Chapter 5.
Scheme Extensions Da thru Iz

Topic: Ignore

ds:debug

Scheme Extension: Deformable Surfaces

Action: Prints parameters of a deformable model.

Filename: adm/ds_scm/dsscm.cxx

APIs: api_dm_get_attrib_dm2acis

Syntax: `(ds:debug owner [target=1])`

Arg Types:

owner	entity
target	integer

Returns: unspecified

Errors: None

Description: Prints a report of the target deformable model parameters belonging to the owner.

The target argument specifies which deformable model to use in a patch hierarchy. Valid values for target are:

1 = active deformable model
2 = root deformable model
–1 = active deformable model and offspring
–2 = root deformable model and offspring

Otherwise, the target is the deformable model whose tag identifier equals target.

owner ACIS face or edge on which the deformable model lives.

target specifies which deformable model to use in a patch hierarchy.

Limitations: None

Example:

```
; ds:debug
; Print parameters of a deformable surface model.
; Build a test square spline face.
; (6x6 control points, x and y side length = 36)
(define dsmodel1 (ds:test-face 6 6 36 36 0))
;; dsmodel1
; Output its parameters
(ds:debug dsmodel1 1)
; Output to stdout
;; Rendering Facets[nu nv]        : [10, 10]  ok
;;  tag_object_count              : [4]
;;  control_pt_count              : [36]
;;  domain_dim,image_dim          : [2, 3]
;; has domain changed since commit?: 0
;;  (0=no,1=yes)
;; stretch_parameters [a11 a12 a22]: [1.00 0.00 1.00]
;; bending_parameters [b11 b12 b22]: [5.00 0.00 5.00]
;; displacement_param [d          ]: [0.00]
;; time_integration [dt mass damp]: [1.00 1.00 5.00]
;;  spatial_integration [ntgrl_degree]: [8]
;;  [number_of_loads, number_of_cstrns]   : [0 4]
;;  ends(0=open,1=closed,2=tang) [u v]: [0 0]
;;  poles(0=no,1=lo,2=hi,3=both) [u v]: [0 0]
;; abcd_state [rebuild/solve A B C D] :[A(1 0)
;;  b(1 0) C(1 0) d(1 0)]
;; abcd_state [dyn_terms init_x]  : Dyn terms(1)
;;  Init_x(0)
;; sym_eq degree_of_freedom counts:
;;  Total_system_dofs                 : n = 36
;;   Total_constraint_equation_count  : q = 0
;;  Free_system_dofs                  : p = 36
;;   Independent_Constraint_Count     : m = 0
;;  Basis-Degrees [Basis_u,Basis_v]   : [3, 3]
; ; Knot-counts [Basis_u,Basis_v] : [4, 4]
;; Basis_u knots ([mult] = val)   : [3] = 0.000000
;;                                : [1] = 0.333333
;;                                : [1] = 0.666667
;;                                : [3] = 1.000000
;; Basis_v knots ([mult] = val)   : [3] = 0.000000
;;                                : [1] = 0.333333
;;                                : [1] = 0.666667
;;                                : [3] = 1.000000
;; ()
```

ds:elevate–degree

Scheme Extension: Deformable Surfaces

Action: Increases the polynomial degree of a deformable model's basis functions.

Filename: adm/ds_scm/dsscm.cxx

APIs: api_dm_get_attrib_dm2acis

Syntax: `(ds:elevate–degree owner [target=1] [continuity–flag=0])`

Arg Types:		
	owner	entity
	target	integer
	continuity–flag	integer

Returns: unspecified

Errors: None

Description: Increments by one the degree of the basis functions of the target deformable model belonging to owner.

The target argument specifies which deformable model to use in a patch hierarchy. Valid values for target are:

1 = active deformable model
2 = root deformable model
–1 = active deformable model and offspring
–2 = root deformable model and offspring

Otherwise, the target is the deformable model whose tag identifier equals target.

The continuity–flag specifies if the element-to-element continuity level is to be preserved or changed when the element degree is incremented. Higher degree elements can support higher element-to-element continuity. When continuity–flag is 0, the inter-element continuity is preserved. When continuity–flag is 1, the inter-element continuity is increased.

Only when continuity–flag is 0 and inter-element continuity is preserved can the returned shape exactly equal the input shape. Changing the inter-element continuity will change the shape by a small amount, which may be surprisingly large for some cases of NURB (or NUB) curves and surfaces.

When cont–flag is 1, one degree of freedom (dof) is added to each axis of the surface. So a 6x6 control point surface will become a 7x7 control point surface. So if a bi–cubic 6x6 control point surface initially composed of an array of 3x3 elements is used as input, then the modified surface is a 9x9 control point surface.

owner ACIS face or edge on which the deformable model lives.

target specifies which deformable model to use in a patch hierarchy.

continuity–flag specifies if the element-to-element continuity level is to be preserved or changed when the element degree is incremented.

Limitations: None

Example:

```
; ds:elevate–degree
; Create a low–order B–spline face and use this
; function to increase the order of its deformable
; model. Make a low–order test face.
; (4x4 control points, x and y side length = 36)
(define dsmodel1 (ds:test–face 6 6 36 36 0))
;; dsmodel1
(define erase (entity:erase dsmodel1))
;; erase
(define drawstate (ds:set–draw–state dsmodel1 2
    ds–draw–cpts))
;; drawstate
; The face should only have 4 (2x2) control points.
; Increase the basis polynomial degree to 3.
(ds:elevate–degree dsmodel1 2)
;; ()
(ds:get–shape–dofs dsmodel1)
;; (81 9 9 (1.73686342001475e–013
; –8.49755921010498e–01
; .
; . (entire return is displayed in Scheme window)
; .
;; 000000000041 ...))
(ds:elevate–degree dsmodel1 2)
;; ()
; The face now has 16 (4x4) control points.
; Use ds:debug to examine the changes.
```

ds:end–sculpting

Scheme Extension: Deformable Surfaces

Action: Removes and deletes the nonpersistent deformable surface model from an entity.

Filename: adm/ds_scm/dsscm.cxx

APIs: api_dm_get_attrib_dm2acis, api_dm_remove_attrib_dm2acis

Syntax: `(ds:end–sculpting owner)`

Arg Types: owner entity

Returns: unspecified

Errors: None

Description: Removes and deletes the nonpersistent version of the specified deformable surface model. This command does not change the owner's geometry or update its persistent version of the deformable model.

owner ACIS face or edge on which the deformable model lives.

Limitations: None

Example:

```
; ds:end–sculpting
; Make a test face.
; (6x6 control points, x and y side length = 36)
(define dsmodel1 (ds:test–face 6 6 36 36 0))
;; dsmodel1
; Don't display entity / ds test face exists.
(define erase (entity:erase dsmodel1))
;; erase
; Render the loads and constraints.
(ds:set–draw–state dsmodel1 1
    (+ ds–draw–cstrns ds–draw–loads))
;; ()
; Force the construction of a deformable model by
; running any DS extensions on the face.
(ds:set–draw–state dsmodel1 1
    (+ ds–draw–cstrns ds–draw–loads))
;; ()
; The deformable face renders.
; Remove the deformable model.
(ds:end–sculpting dsmodel1)
;; ()
(view:refresh)
;; #[view 1076131480]
; The deformable model no longer renders.
```

ds:extrapolate

Scheme Extension: Deformable Surfaces

Action: Expands the owner's deformable mode by 5% in all possible directions.

Filename: adm/ds_scm/dsscm.cxx

APIs: api_dm_get_attrib_dm2acis

Syntax: `(ds:extrapolate owner target=1)`

Arg Types:

owner	entity
target	integer

Returns: unspecified

Errors: None

Description: Extends the domain of the target deformable model by 5% in all possible directions.

The target argument specifies which deformable model to use in a patch hierarchy. Valid values for target are:

1 = active deformable model
2 = root deformable model
–1 = active deformable model and offspring
–2 = root deformable model and offspring

Otherwise, the target is the deformable model whose tag identifier equals target.

This extension recalculates the control point locations so that the original portion of the surface maintains its current shape. Faces with closed, periodic, or singular edges cannot be extended in those directions but are extended in directions across edges which are not so constrained.

A deformable model within a multi–surface mesh will not be extrapolated. Trying to do so will not change the database in any manner and result in signaling the system error DM_MULTI_SURF_EXTRAPOLATE.

owner ACIS face or edge on which the deformable model lives.

target specifies which deformable model to use in a patch hierarchy.

Limitations: Can’t extrapolate a deformable model within a multi–surface mesh.

Example:

```
; ds:extrapolate
; Extends the domain of the surface by 5% in each
; possible direction.
; Build a test square spline face.
; (6x6 control points, x and y side length = 36)
(define dsmodel1 (ds:test–face 6 6 36 36 0))
;; dsmodel1
; Don’t display entity / ds test face exists.
(define erase (entity:erase dsmodel1))
;; erase
; Render the loads and constraints.
(ds:set–draw–state dsmodel1 1
    (+ ds–draw–cstrns ds–draw–loads))
;; ()
; Add a pt–cstrn at the center and track it.
(define cc1 (ds:add–pt–cstrn dsmodel1
    1 ”position” (par–pos 0.5 0.5)))
;; cc1
(ds:set–pt–xyz dsmodel1 cc1 0
    (position 18 18 35))
;; 8
; Compute a new deformable model position.
(ds:solve dsmodel1 1 1)
;; ()
; OUTPUT Original

; Extrapolate the surface by 5%.
(ds:extrapolate dsmodel1 1)
;; ()
(ds:solve dsmodel1 1 1)
;; ()
; The shape remains the same.
; OUTPUT Result
```

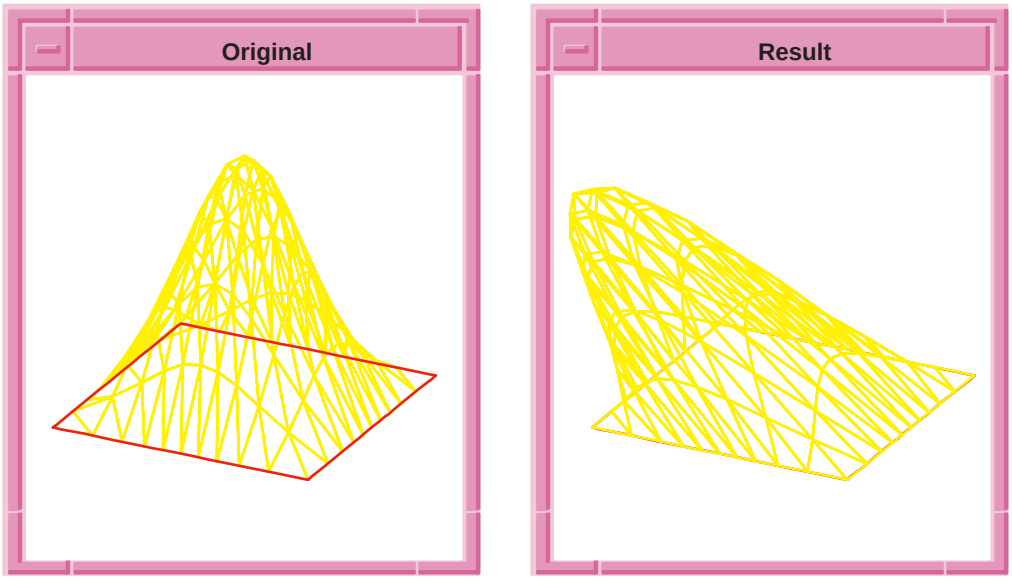


Figure 5-1. ds:extrapolate

ds:gen–path

Scheme Extension: Deformable Surfaces

Action: Creates a list of sampled positions and time stamp for each position taken from the trajectory of a flying particle.

Filename: adm/ds_scm/dsscm.cxx

APIs: None

Syntax:
```
(ds:gen–path [start–time=0 stop–time=3600
store–count=121
    integration–count=3])
```

Arg Types:		
	start–time	real
	stop–time	real
	store–count	integer
	integration–count	integer

Returns: integer

Errors: None

Description: Creates a list of regularly sampled positions generated through a time integration of the equations of motion for a powered particle. This function is only a convenience for making test–cases to be used in the construction of deformable curve spring–set loads.

start–time is the starting time in seconds (default 0.0). stop–time is the stop time in seconds (default 3600). store–count is the stored time step count; The default is 121. integration–count is the integration steps per store; The default is 3.

This returns a list consisting of an integer for the point count, a list of time values, and a list of positions.

start–time is the starting time in seconds.

stop–time is the stop time in seconds.

store–count is the stored time step count.

integration–count is the integration steps per store.

Limitations: None

Example:

```
; ds:gen–path
; Use a set of point locations defined by the
; function (ds:gen–path) to mold the shape of
; of a deformable curve with a spring–set load.
; Use ds:gen–path to make positions and time samples.
(define samp–pts1 (ds:gen–path 0 650 121 3))
;; samp–pts1
; Build a deformable curve.
(define dsmodel1
    (ds:test–edge 25 36 0 0 3 0.0 0.0 650))
;; dsmodel1
; Don't display entity / ds test face exists.
(define erase (entity:erase dsmodel1))
;; erase
; Add a spring set to the deformable curve
; domain–points list, position points list,
; gain.
(define spr1 (ds:add–spring–set dsmodel1 1
    (cadr samp–pts1) (caddr samp–pts1) 100))
;; spr1
; Toggle the end point constraints off so the ends
; can move to the sampled data.
(ds:toggle–cstrn dsmodel1 1)
```

```
;; 0
(ds:toggle–cstrn dsmodel1 2)
;; 0
; Clear the deformable curve’s default shape.
(ds:set–default–shape dsmodel1 1 0)
;; ()
; Render cstrns, loads, curvature combs, elements.
(ds:set–draw–state dsmodel1 1
    (+ ds–draw–cstrns ds–draw–loads
    ds–draw–curve–comb ds–draw–elems))
;; ()
(ds:set–comb–graphics dsmodel1 1 25 –100)
;; ()
(ds:set–icon–radius dsmodel1 3)
;; ()
; call (ds:solve) so that data may sculpt curve.
(ds:solve dsmodel1 1 1)
;; ()
; Capture the shape without the data points using
; the default shape feature.
(ds:set–default–shape dsmodel1 1 1)
;; ()
(ds:rm–tag–object dsmodel1 spr1)
;; 5
(ds:set–draw–state dsmodel1 1
    (+ ds–draw–cstrns ds–draw–loads
    ds–draw–curve–comb))
;; ()
(ds:solve dsmodel1 1 1)
;; ()
; Captured curve shape is drawn with curvature comb.
```

ds:get–active–patch

Scheme Extension: Deformable Surfaces

Action: Gets the tag identifier number of the currently active patch within the patch hierarchy of the given deformable model.

Filename: adm/ds_scm/dsscm.cxx

APIs: api_dm_get_attrib_dm2acis

Syntax: (**ds:get–active–patch** owner)

Arg Types:	owner	entity

Returns: integer

Errors: None

Description: Returns the tag identifier number for the active patch within the deformable model patch hierarchy associated with the input owner.

Making the query on an owner without a deformable model causes a deformable model to be added to the entity and a tag identifier value is returned.

owner ACIS face or edge on which the deformable model lives.

Limitations: None

Example:

```
; ds:get-active-patch
; Add to a square test face a parabolic crv-load
; and use it.
; Build a test square spline face.
; (6x6 control points, x and y side length = 36)
(define dsmodel1 (ds:test-face 6 6 36 36 0))
;; dsmodel1
; Don't display entity / ds test face exists
(define erase (entity:erase dsmodel1))
;; erase
; Render the loads and constraints.
(ds:set-draw-state dsmodel1 1
    (+ ds-draw-cstrns ds-draw-loads))
;; ()

; Add a pt-cstrn at the center of the parent and
; track it.
(define cc1 (ds:add-pt-cstrn dsmodel1
    1 "position" (par-pos 0.5 0.5)))
;; cc1
(ds:set-pt-xyz dsmodel1 cc1 0
    (position 18 18 10))
;; 8
; Compute a new deformable model position
(ds:solve dsmodel1 1 1)
;; ()

; Add a patch to the parent shape.
; The new patch becomes the active shape.
(define patch1 (ds:add-patch dsmodel1 1 1
    (par-pos 0.3 0.3) (par-pos 0.5 0.5)
    (par-pos 0.7 0.7) 3))
;; patch1
```

```
; Add and track a pt–cstrn on the patch.
(define cc2 (ds:add–pt–cstrn dsmodel1
    1 ”position” (par–pos 0.5 0.5)))
;; cc2
(ds:set–pt–xyz dsmodel1 cc2 0
    (position 18 18 36))
;; 8
; Compute a new deformable model position
(ds:solve dsmodel1 1 –3)
;; ()

; Find the tag identifier value of the active patch
(ds:get–active–patch dsmodel1)
;; 8
```

ds:get–alpha

Scheme Extension: Deformable Surfaces

Action: Gets a list of (au, av, atheta) for deformable surfaces and a list of (au) for deformable curves.

Filename: adm/ds_scm/dsscm.cxx

APIs: api_dm_get_attrib_dm2acis

Syntax: (**ds:get–alpha** owner [target=1])

Arg Types:

owner	entity
target	integer

Returns: (real ...)

Errors: None

Description: Returns a list of (au, av, atheta) surfaces and a list of (au) for deformable curves. These are the owner and target resistance to stretch parameters.

au is the resistance to stretch in the parametric *u* direction and av is the resistance to stretch in the parametric *v* direction. When au = av, the surface has homogeneous properties. The resistance to shape is the same in all parametric directions. When au is not equal to av, the inhomogeneous material property behavior is rotated within the surface by the angle, atheta, given in degrees.

The target argument specifies which deformable model to use in a patch hierarchy. Valid values for target are:

1 = active deformable model
2 = root deformable model
–1 = active deformable model and offspring
–2 = root deformable model and offspring

Otherwise, the target is the deformable model whose tag identifier equals target.

owner ACIS face or edge on which the deformable model lives.

target specifies which deformable model to use in a patch hierarchy.

Limitations: None

Example:

```
; ds:get–alpha
; Build a test square spline face.
; (6x6 control points, x and y side length = 36)
(define dsmodel1 (ds:test–face 6 6 36 36 0))
;; dsmodel1
; Don't display entity / ds test face exists.
(define erase (entity:erase dsmodel1))
;; erase
; Render the loads and constraints.
(ds:set–draw–state dsmodel1 1
    (+ ds–draw–cstrns ds–draw–loads))
;; ()
; Ensure that the dof state is current.
(ds:solve dsmodel1 1)
;; ()
; Get the alpha state for the surface.
(ds:get–alpha dsmodel1 1)
;; (1 1 0)
```

ds:get–beta

Scheme Extension: Deformable Surfaces

Action: Gets a list of (bu, bv, btheta) for deformable surfaces and returns a list of (bu) for deformable curves.

Filename: adm/ds_scm/dsscm.cxx

APIs: api_dm_get_attrib_dm2acis

Syntax: `(`**`ds:get–beta`** `owner [target=1])`

Arg Types:		
	owner	entity
	target	integer

Returns: (real ...)

Errors: None

Description: Returns a list of (bu, bv, btheta) for deformable surfaces and returns a list of (bu) for deformable curves. These are the owner's target deformable model's resistance to bending parameters.

The target argument specifies which deformable model to use in a patch hierarchy. Valid values for target are:

1 = active deformable model
2 = root deformable model
–1 = active deformable model and offspring
–2 = root deformable model and offspring

Otherwise, the target is the deformable model whose tag identifier equals target.

In deformable surfaces, bu is the resistance to stretch in the parametric *u* direction and bv is the resistance to stretch in the parametric *v* direction. When bu equals bv, the surface has homogeneous properties. The resistance to shape is the same in all parametric directions. When bu does not equal bv the surface's inhomogeneous material property behavior is rotated within the surface by the angle, btheta, given in degrees.

owner ACIS face or edge on which the deformable model lives.

target specifies which deformable model to use in a patch hierarchy.

Limitations: None

Example:

```
; ds:get–beta
; Build a test square spline face.
; (6x6 control points, x and y side length = 36)
(define dsmodel1 (ds:test–face 6 6 36 36 0))
;; dsmodel1
; Don’t display entity / ds test face exists.
(define erase (entity:erase dsmodel1))
;; erase
; Render the loads and constraints.
(ds:set–draw–state dsmodel1 1
    (+ ds–draw–cstrns ds–draw–loads))
;; ()
; Ensure the dof state is current.
(ds:solve dsmodel1 1)
;; ()
; Get the beta state for the surface.
(ds:get–beta dsmodel1 1)
;; (5 5 0)
```

ds:get–child–tag

Scheme Extension: Deformable Surfaces, DML Tags

Action: Returns the tag number of the target deformable model’s child or –1 for none.

Filename: adm/ds_scm/dsscm.cxx

APIs: api_dm_get_attrib_dm2acis

Syntax: (**ds:get–child–tag** owner [target=1])

Arg Types:

owner	entity
target	integer

Returns: integer

Errors: None

Description: Returns the tag identifier for the target deformable model’s child or –1 for none.

The target argument specifies which deformable model to use in a patch hierarchy. Valid values for target are:

1 = active deformable model
2 = root deformable model

Otherwise, the target is the deformable model whose tag identifier equals target.

owner ACIS face or edge on which the deformable model lives.

target specifies which deformable model to use in a patch hierarchy.

Limitations: None

Example:

```
; ds:get–child–tag
; Build a test square spline face.
; (6x6 control points, x and y side length = 36)
(define dsmodel1 (ds:test–face 6 6 36 36 0))
;; dsmodel1
; Don’t display entity / ds test face exists
(define erase (entity:erase dsmodel1))
;; erase
; Render the loads and constraints.
(ds:set–draw–state dsmodel1 1
    (+ ds–draw–cstrns ds–draw–loads))
;; ()

; Add two rectangular patches to the child shape.
; The new patch becomes the active shape.
(define patch1 (ds:add–patch dsmodel1 2 1
    (par–pos 0.1 0.1) (par–pos 0.3 0.3)
    (par–pos 0.5 0.5) 3))
;; patch1
(define patch2 (ds:add–patch dsmodel1 2 1
    (par–pos 0.7 0.7) (par–pos 0.8 0.8)
    (par–pos 0.9 0.9) 3))
;; patch2

; exercise ds:get–child–tag
(ds:get–child–tag dsmodel1 2)
;; 7
```

ds:get–comb–graphics

Scheme Extension: Deformable Surfaces

Action: Gets the number of points and the gain used in the rendering of a curvature comb graphical report for a deformable model.

Filename: adm/ds_scm/dsscm.cxx

APIs: api_dm_get_attrib_dm2acis

Syntax: **(ds:get–comb–graphics** owner [target=1])

Arg Types:

owner	entity
target	integer

Returns: ((integer . real) ...)

Errors: None

Description: Gets the number of points and scaling gain used in the graphical parameter space curvature comb report for a target deformable model.

The target argument specifies which deformable model to use in a patch hierarchy. Valid values for target are:

1 = active deformable model
2 = root deformable model
–1 = active deformable model and offspring
–2 = root deformable model and offspring

Otherwise, the target is the deformable model whose tag identifier equals target.

Curvature combs are a graphical representation of the parameter space curvature properties of the curve constraints. A curvature comb is made up of a set of vectors drawn from the curve in the direction of principle curvature. The magnitude of the vector is the curvature. These combs are then projected through the deformable surface shape into three dimensional space for rendering which tends to distort them as the surface is manipulated.

The number of vectors drawn per element is specified by elem_pt_count. gain is an additional gain factor used to scale the curvature comb for viewing convenience.

owner ACIS face or edge on which the deformable model lives.

target specifies which deformable model to use in a patch hierarchy.

Limitations: None

Example:

```
; ds:get–comb–graphics
; Build a test square face with some tag objects
; (6x6 control points, x and y side length = 36)
(define dsmodel1 (ds:test–face 6 6 36 36 0))
;; dsmodel1
; Don’t display entity / ds test face exists.
(define erase (entity:erase dsmodel1))
;; erase
; Render the loads and constraints.
(ds:set–draw–state dsmodel1 1
    (+ ds–draw–cstrns ds–draw–loads))
;; ()
; Replace the default edge constraints with a circle
; constraint.
(ds:toggle–cstrn dsmodel1 1)
;; 0
(ds:toggle–cstrn dsmodel1 2)
;; 0
(ds:toggle–cstrn dsmodel1 3)
;; 6
(ds:toggle–cstrn dsmodel1 4)
;; 6
(ds:add–circ–cstrn dsmodel1 1 ”position”
    (par–pos 0.5 0.5) (par–pos 0 0.3)
    (par–pos 0.3 0))
;; 7
; Render the loads, constraints, and the
; curvature reports.
(ds:get–draw–state dsmodel1 1)
;; 12
; Adjust the curvature plot graphics
; to draw the comb pointing to the convex side.
(ds:set–comb–graphics dsmodel1 1 20 –1.0)
;; ()
; Retrieve the curvature plot graphical parameters.
(ds:get–comb–graphics dsmodel1 1)
;; (20 –1)
```

ds:get–cstrn

Scheme Extension: Deformable Surfaces

Action: Gets the defining data, current behavior, and current state of a deformable model constraint.

Filename: adm/ds_scm/dsscm.cxx

APIs: api_dm_get_attrib_dm2acis

Syntax: `(ds:get-cstrn owner tag)`

Arg Types:

owner	entity
tag	integer

Returns: string | string ...

Errors: None

Description: This extension retrieves all defining and current status data for an existing constraint in a deformable model. The input owner argument identifies the face or edge being sculpted. The tag identifier specifies which constraint to query. When the tag identifier recognizes a constraint, the deformable model in the patch hierarchy which contains the constraint is made the active deformable model. The constraint data is organized and returned as a list whose member order and allowed values are described below. The list of return items includes:

```
( patch1_tag
      patch2_tag
      behavior_string
      state_string
      rights_string
      shape1_string
      (par_pos list)
      shape2_string
      (par_pos list)
      tangent_gain_value
      integral_degree_accuracy
      )
```

patch_tag is the tag identifier of the patch that contains given tag object or –1 when the input tag does not identify a valid tag object.

behavior specifies how the constraint constrains the position and/or the cross–tangent. The behavior strings are the same for point and curve constraints, but different for link constraints. For point and curve constraints, the output is a combination of the strings, “position”, “tangent”, and “curvature”. The presence on any string specifies the properties being constrained by the constraint. The behavior string for a link constraint reports the position and the cross tangent state for each curve in the following form:

pos_“crv1_pos”_“crv2_pos”_tan_“crv1_tan”_“crv2_tan”

The “crv_behavior” strings are one of: “link”, “fix”, or “off”. Where “link” means the two curves are tracking one another, “fix” means the curve is fixed and acting like a curve constraint, and “off” means the curve is unconstrained and off to deform as desired. Typical link behavior strings will be “pos_link_link_tan_off_off”, for a C0 link constraint, “pos_link_link_tan_link_link” for a C1 link constraint, and “pos_fix_fix_tan_link_link” for a C1 link constraint where the surface is only off to rotate about the constraint’s own axis.

pos_link_link_tan_off_off	for a C0 link constraint
pos_link_link_tan_link_link	for a C1 link constraint and
pos_fix_fix_tan_link_link	for a C1 link constraint where the surface is only able to rotate about the constraint’s own axis.

state specifies whether the constraint is currently on or off. The state argument is a string whose value is either “on” or “off”.

rights specifies whether the constraint may be turned off and whether the constraint may be deleted by the user interface. For example, constraints preserving boundary shapes to fit into solid models which cannot be deleted or stopped are immutable. The valid values for rights are:

– “deletable”
– “stopable”
– “delete_stopable”
– “immutable”

shape specifies which loci of points in the deformable model are constrained. The supported constraints include point constraints and curve constraints in the form of straights, parabolas, circ, and general pcurves. A circ may be used for either an ellipse or a circle. The valid values for shape are:

– “point”
– “straight”
– “parabola”
– “circ”
– “curve”
– “area”

The par–pos lists are one or more par_pos point locations that parameterize a shape. Each par_pos location specifies a point in the domain space of the deformable model. The coordinates of any par–pos point are scaled to the range of 0.0 to 1.0. Different shapes contain different numbers of par–pos point positions as follows:

Shape	uv_pts Count	uv_pts Description
“straight”	2	has two par–pos positions: Begin *uv* point and end *uv* point.
“parabola”	3	has three par–pos positions: Begin *uv* point, tangent intersection point, and end *uv* point.
“circ”	3	has three par–pos positions: Center *uv* point, *a* axis end *uv* point, and *b* axis end *uv* point.
“area”	2	has two par–pos positions: lower–left and upper–right corners of a rectangle area. Currently, all areas are limited to simple rectangles.
tag	0	has no par–pos positions. The shape is taken from an existing load and the load is deleted.

“circ” is a closed elliptical arc defined by a center point (*uv*_ctr), and two vectors (*uv*_a and *uv*_b) which mark the distance between the center point and two points on the ellipse. The shape of the curve in domain space is given by:

```
C (theta) = uv_ctr +
            uv_a*cos(theta)
            + uv_b*sin(theta)
```

When the *a* and *b* vectors have the same lengths the “circ” is a circle centered on the given center point. A well paramaterized “circ” is best built when the *a* and *b* vectors are orthogonal to one another. For example:

```
uv_ctr  = [.5,.5], uv_a
        = [.2,0], uv_b = [0,.2]
```

tang_gain is a scaling value for the tangent vectors constrained in a curve’s tangent constraint.

integral_degree specifies the accuracy of numerical integration used within each element. (A polynomial function of degree integral_degree will be integrated exactly.) An integral_degree value that is too large increases the computation cost and reduces the error, but an integral degree value that is too small yields bad results. The integral degree value is selected automatically by the deformable modeling library. For the constraints which do not use integral_degree, point–constraints, and area–constraints, set values to –1.

owner ACIS face or edge on which the deformable model lives.

tag identifier recognizes a constraint.

Limitations: None

Example:

```
; ds:get–cstrn
; Add to a square test face a parabolic crv–load
; and use it.
; Build a test square spline face.
; (6x6 control points, x and y side length = 36)
(define dsmodel1 (ds:test–face 6 6 36 36 0))
;; dsmodel1
; Don't display entity / ds test face exists.
(define erase (entity:erase dsmodel1))
;; erase
; Render the loads and constraints.
(ds:set–draw–state dsmodel1 1
    (+ ds–draw–cstrns ds–draw–loads))
;; ()
; Add a pt–cstrn at the center of the parent and
; track it.
(define cc1 (ds:add–cstrn dsmodel1
    1 "point" "position" (par–pos 0.5 0.5)))
;; cc1
(ds:set–pt–xyz dsmodel1 cc1 0 (position 16 16 10))
;; 8
; Compute a new deformable model position.
(ds:solve dsmodel1 1 1)
;; ()
(ds:get–cstrn dsmodel1 cc1)
;; (2 –1 2 "position" "on" "delete_stopable" "point"
;; (#[par–pos 0.5 0.5]) "none" () 1 –1)
```

ds:get–default–state

Scheme Extension: Deformable Surfaces

Action: Gets the default shape state, which is 0 when not being used, else returns 1.

Filename: adm/ds_scm/dsscm.cxx

APIs: api_dm_get_attrib_dm2acis

Syntax: `(ds:get–default–state owner [target=1])`

Arg Types:

owner	entity
target	integer

Returns: integer

Errors: None

Description: Gets the face's deformable model default shape state. When the return flag is 0, the default state is zeroed. When the return flag is 1, the entity is using a default shape.

The target argument specifies which deformable model to use in a patch hierarchy. Valid values for target are:

1 = active deformable model
2 = root deformable model
–1 = active deformable model and offspring
–2 = root deformable model and offspring

Otherwise, the target is the deformable model whose tag identifier equals target.

The default state is the state to which the surface is attempting to deform. The surface always deforms to the default state when its shape is computed without any loads or constraints.

Objects, like soap film for example, attempt to minimize their area. Their default state is a zero area flat element. The state of such deformable models is completely determined by the applied constraints and loads. This behavior can be mimicked in deformable surfaces by setting the default state to zero. Default states tend to be very smooth.

Most objects, like steel plates, rubber balls, and plastic baubles, have a natural default state. Loads and constraints pull the object away from its natural state. This behavior may be mimicked by capturing a non–zero default state with the command ds:set–default–shape.

owner ACIS face or edge on which the deformable model lives.

target specifies which deformable model to use in a patch hierarchy.

Limitations: None

Example:

```
; ds:get–default–state
; Build a test square face with some tag objects
; (6x6 control points, x and y side length = 36)
(define dsmodel1 (ds:test–face 6 6 36 36 0))
;; dsmodel1
; Don’t display entity / ds test face exists.
(define erase (entity:erase dsmodel1))
;; erase
; Render the loads and constraints.
(ds:set–draw–state dsmodel1 1
    (+ ds–draw–cstrns ds–draw–loads))
;; ()
(ds:solve dsmodel1 1 1)
;; ()
; The control points don’t move because
; the surface is created with a default shape.
(ds:get–default–state dsmodel1)
;; 1
; Replace the default edge constraints with corner
; point constraints.
(ds:toggle–cstrn dsmodel1 1)
;; 0
(ds:toggle–cstrn dsmodel1 2)
;; 0
(ds:toggle–cstrn dsmodel1 3)
;; 6
(ds:toggle–cstrn dsmodel1 4)
;; 6
(ds:add–pt–cstrn dsmodel1 1 ”position”
    (par–pos 0 0))
;; 7
(ds:add–pt–cstrn dsmodel1 1 ”position”
    (par–pos 1 0))
;; 8
(ds:add–pt–cstrn dsmodel1 1 ”position”
    (par–pos 0 1))
;; 9
(ds:add–pt–cstrn dsmodel1 1 ”position”
    (par–pos 1 1))
;; 10
```

```
; Eliminate the default state behavior.
(ds:set–default–shape dsmodel1 1 0)
;; ()
(ds:get–delta dsmodel1)
;; 0
(ds:solve dsmodel1 1 1)
;; ()
; Control points relax to a lower energy position.
(ds:get–default–state dsmodel1)
;; 0
```

ds:get–delta

Scheme Extension: Deformable Surfaces

Action: Gets a list of the owner's deformable model's resistance to displacement.

Filename: adm/ds_scm/dsscm.cxx

APIs: api_dm_get_attrib_dm2acis

Syntax: (**ds:get–delta** owner [target=1])

Arg Types:		
	owner	entity
	target	integer

Returns: (real ...)

Errors: None

Description: For the given owner, returns the target deformable model's resistance to displacement from the default shape parameter value for the target deformable model.

The target argument specifies which deformable model to use in a patch hierarchy. Valid values for target are:

1 = active deformable model
2 = root deformable model
–1 = active deformable model and offspring
–2 = root deformable model and offspring

Otherwise, the target is the deformable model whose tag identifier equals target.

owner ACIS face or edge on which the deformable model lives.

target specifies which deformable model to use in a patch hierarchy.

Limitations: None

Example:

```
; ds:get–delta
; Build a test square spline face.
; (6x6 control points, x and y side length = 36)
(define dsmodel1 (ds:test–face 6 6 36 36 0))
;; dsmodel1
; Don't display entity / ds test face exists.
(define erase (entity:erase dsmodel1))
;; erase
; Render the loads and constraints.
(ds:set–draw–state dsmodel1 1
    (+ ds–draw–cstrns ds–draw–loads))
;; ()
; Ensure that the dof state is current.
(ds:solve dsmodel1 1 1)
;; ()
; Get the alpha state for the surface.
(ds:get–delta dsmodel1 1)
;; 0
; Specify the surface's resistance to
; displacement parameter.
(ds:set–delta dsmodel1 1 0.0)
;; ()
; Get the delta state for the surface.
(ds:get–delta dsmodel1 1)
;; 0
```

ds:get–dmod–tags

Scheme Extension: Deformable Surfaces

Action: This extension returns the tags in a single deformable model.

Filename: adm/ds_scm/dsscm.cxx

APIs: api_dm_get_attrib_dm2acis

Syntax: (**ds:get–tags owner** owner tag)

Arg Types:

owner	entity
tag	integer

Returns: integer ...

Errors: None

Description: Returns an integer list of tags corresponding to all tag objects in the selected deformable model.

owner ACIS face or edge on which the deformable model lives.

tag identifier recognizes a constraint.

Limitations: None

Example:
```
; ds:get-dmod-tags
; No example available at this time
```

ds:get–dmods

Scheme Extension: Deformable Surfaces

Action: This extension returns all the DS_dmods in a deformable modeling hierarchy.

Filename: adm/ds_scm/dsscm.cxx

APIs: api_dm_get_attrib_dm2acis

Syntax: `(ds:get-dmods owner)`

Arg Types:	owner	entity

Returns: integer

Errors: None

Description: A deformable modeling hierarchy can contain more than 1 DS_dmod, for example child patches and multi-surface DS_dmods. This routine returns all DS_dmods in the hierarchy.

owner ACIS face or edge on which the deformable model lives.

Limitations: None

Example:
```
; ds:get-dmods
; No example available at this time
```

ds:get–dof–state

Scheme Extension: Deformable Surfaces

Action: Gets the list of the owner's degree of freedom count for the deformable surface.

Filename: adm/ds_scm/dsscm.cxx

APIs: api_dm_get_attrib_dm2acis

Syntax: `(ds:get-dof-state owner [target=1])`

Arg Types:

owner	entity
target	integer

Returns: (integer ...)

Errors: None

Description: Returns the list of the owner's degree of freedom count for the target deformable surface. The list is ordered as total_dof_count, free_dof_count, total_constraint_count, fixed_count.

where:

total_dof_count	is the total number of surface degrees of freedom (the control point count).
free_dof_count	is the number of degrees of freedom left to deform after applying constraints. If this number gets too low due to lots of constraints the surface will no longer be able to deform in a natural manner.
total_constraint_count	is the total number of constraint equations. Many of these will be redundant constraints.
fixed_count	is the number of independent constraints found within the set of constraint equations. Each independent constraint removes one degree of freedom from the free_count.

total_dof_count = free_dof_count + fixed_dof_count

The target argument specifies which deformable model to use in a patch hierarchy. Valid values for target are:

1 = active deformable model
2 = root deformable model
–1 = active deformable model and offspring
–2 = root deformable model and offspring

Otherwise, the target is the deformable model whose tag identifier equals target.

The deformable surface degree of freedom state is updated after every call to ds:solve. This state information is out of date as soon as any constraints are added or removed from the surface.

owner ACIS face or edge on which the deformable model lives.

target specifies which deformable model to use in a patch hierarchy.

Limitations: None

Example:

```
; ds:get–dof–state
; Build a test square spline face.
; (6x6 control points, x and y side length = 36)
(define dsmodel1 (ds:test–face 6 6 36 36 0))
;; dsmodel1
; Don't display entity / ds test face exists
(define erase (entity:erase dsmodel1))
;; erase
; Render the loads and constraints.
(ds:set–draw–state dsmodel1 1
    (+ ds–draw–cstrns ds–draw–loads))
;; ()
; Ensure that the dof state is current
(ds:solve dsmodel1 1)
;; ()
; Get the dof state for the surface
(ds:get–dof–state dsmodel1 1)
;; (36 36 0 0 0 0)
```

ds:get–draw–grid

Scheme Extension: Deformable Surfaces

Action: Gets the number of mesh polygons used to render the deformable shape of a deformable model.

Filename: adm/ds_scm/dsscm.cxx

APIs: api_dm_get_attrib_dm2acis

Syntax: (**ds:get–draw–grid** owner [target=1])

Arg Types:		
	owner	entity
	target	integer

Returns: integer | integer ...

Errors: None

Description: Gets the number of polygons in the mesh used to render the deformable shape of a target deformable model, where:

nu = number of grid polygons in the v parametric direction
nv = number of grid polygons in the v parametric direction

The target argument specifies which deformable model to use in a patch hierarchy. Valid values for target are:

1 = active deformable model
2 = root deformable model
–1 = active deformable model and offspring
–2 = root deformable model and offspring

Otherwise, the target is the deformable model whose tag identifier equals target.

For deformable surfaces returns (*nu nv*) and for deformable curves returns (*nu*).

owner ACIS face or edge on which the deformable model lives.

target specifies which deformable model to use in a patch hierarchy.

Limitations: None

Example:

```
; ds:get–draw–grid
; Add to a square test face a parabolic crv–load
; and use it.
; Build a test square spline face.
; (6x6 control points, x and y side length = 36)
(define dsmodel1 (ds:test–face 6 6 36 36 0))
;; dsmodel1
; Don't display entity / ds test face exists.
(define erase (entity:erase dsmodel1))
;; erase
; Render the loads and constraints.
(ds:set–draw–state dsmodel1 1
    (+ ds–draw–cstrns ds–draw–loads))
;; ()
(ds:set–draw–grid dsmodel1 1 3 3)
;; ()
(ds:get–draw–grid dsmodel1 1)
;; (3 3)
```

ds:get–draw–state

Scheme Extension: Deformable Surfaces

Action: Gets the combination of deformable model data to be rendered for a deformable model.

Filename: adm/ds_scm/dsscm.cxx

APIs: api_dm_get_attrib_dm2acis

Syntax: `(ds:get–draw–state owner [target=1] flag)`

Arg Types:		
	owner	entity
	target	integer
	flag	integer

Returns: integer

Errors: None

Description: Gets the combination of deformable model data being rendered for the target deformable model.

The argument flag is a bitwise of the following values to display control points, gauss points (used for numerical integration), constraints, and loads.

ds–draw–cpts	draw control points bit
ds–draw–seams	draw seam constraints bit
ds–draw–cstrns	draw constraints bit
ds–draw–loads	draw loads bit
ds–draw–curve–comb	draw curve curvature comb bit
ds–draw–elems	draw element boundaries bit

The following call will cause the constraints, seams, loads, and tangents for the active deformable model to be drawn:

```
(ds:set–draw–state ds–model 1 (+   ds–draw–cstrn
                                   ds–draw–seams
                                   ds–draw–loads
                                   ds–draw–cstrn–norms))
```

The target argument specifies which deformable model to use in a patch hierarchy. Valid values for target are:

1 = active deformable model
2 = root deformable model
–1 = active deformable model and offspring
–2 = root deformable model and offspring

Otherwise, the target is the deformable model whose tag identifier equals target.

owner ACIS face or edge on which the deformable model lives.

target specifies which deformable model to use in a patch hierarchy.

flag is a bitwise of the following values to display control points, gauss points (used for numerical integration), constraints, and loads.

Limitations: None

Example:

```
; ds:get-draw-state
; Build a test square face with some tag objects
; (6x6 control points, x and y side length = 36)
(define dsmodel1 (ds:test-face 6 6 36 36 0))
;; dsmodel1
; Don't display entity / ds test face exists
(define erase (entity:erase dsmodel1))
;; erase
; Render the constraints and loads.
(ds:set-draw-state dsmodel1 1
    (+ ds-draw-cstrns ds-draw-loads))
;; ()
; Render the control points
(ds:set-draw-state dsmodel1 1 ds-draw-cpts)
;; ()
; Get model data.
(ds:get-draw-state dsmodel1 1)
;; 1
; Render the control points and the constraints
(ds:set-draw-state dsmodel1 1
    (+ ds-draw-cpts ds-draw-cstrns))
;; ()
; Get model data.
(ds:get-draw-state dsmodel1 1)
;; 5
; Render the constraints and loads.
(ds:set-draw-state dsmodel1 1
    (+ ds-draw-cstrns ds-draw-loads))
;; ()
; Get model data.
(ds:get-draw-state dsmodel1 1)
;; 12
```

ds:get–dynamics

Scheme Extension: Deformable Surfaces

Action: Gets the time integral data for a deformable surface model time simulation including effective mass, damping, and time step size.

Filename: adm/ds_scm/dsscm.cxx

APIs: api_dm_get_attrib_dm2acis

Syntax: `(ds:get–dynamics owner [target=1])`

Arg Types:

owner	entity
target	integer

Returns: (real . real . real)

Errors: None

Description: Gets the time step size and effective mass and damping for a face's deformable surface.

The target argument specifies which deformable model to use in a patch hierarchy. Valid values for target are:

1 = active deformable model
2 = root deformable model
–1 = active deformable model and offspring
–2 = root deformable model and offspring

Otherwise, the target is the deformable model whose tag identifier equals target.

Each iteration of ds:solve face integrates the equations of motion for a deformable surface by one time step. Large time step values leap from one equilibrium position to another. Small time step values can be used to show the system moving dynamically between equilibrium positions. Increasing the amount of damping increases the rate at which energy is taken from the system. Very large damping values will cause the system to move very slowly. Very small damping values will cause the system to ring. Increasing the system's mass will increase its tendency to ring, i.e. overshoot and bounce back. These effects can be used to create realistic time–based simulations of moving deformable objects.

owner ACIS face or edge on which the deformable model lives.

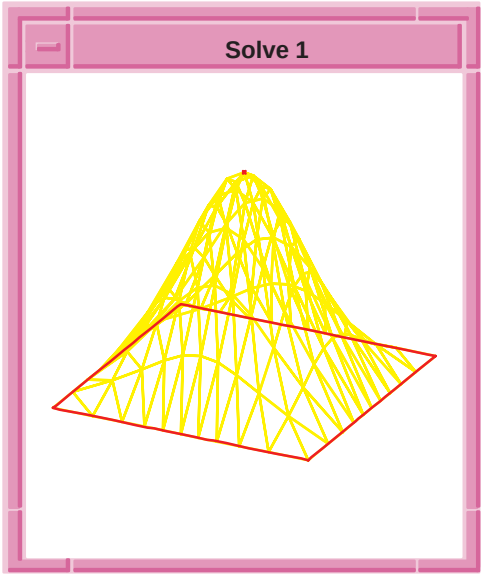
target specifies which deformable model to use in a patch hierarchy.

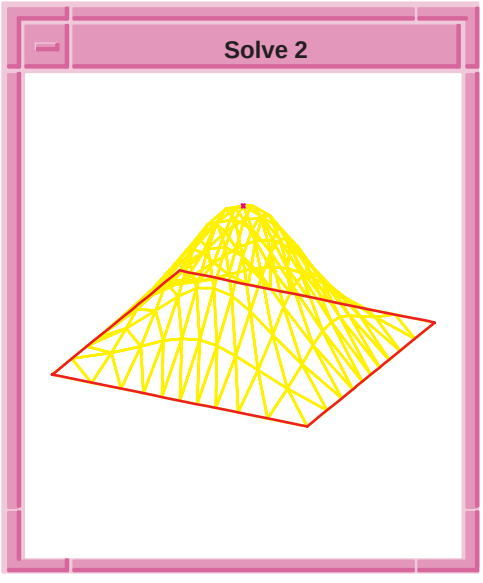
Limitations: None

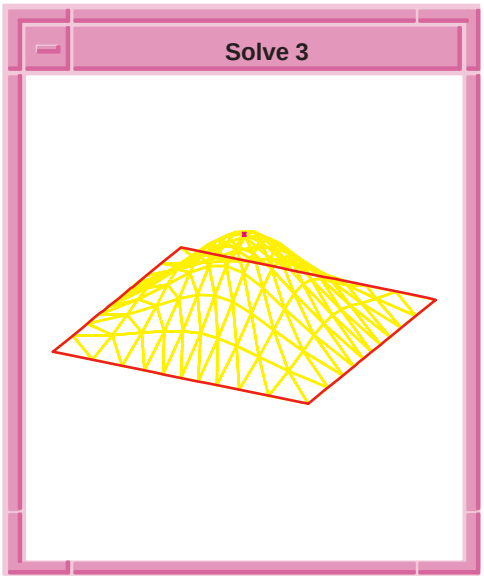
Example:

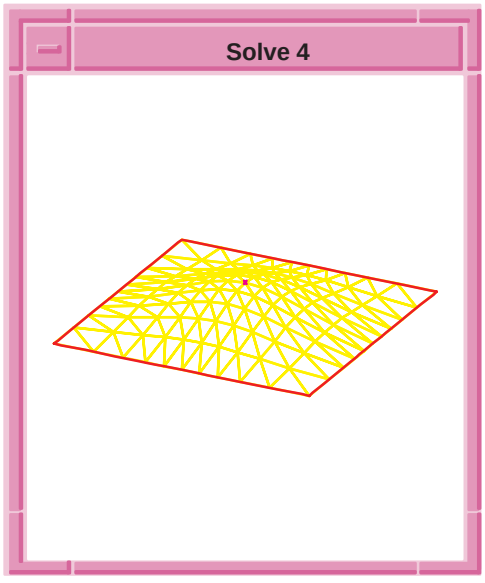
```
; ds:get–dynamics
; Build a test square face with some tag objects
; (6x6 control points, x and y side length = 36)
(define dsmodel1 (ds:test–face 6 6 36 36 0))
;; dsmodel1
; Don’t display entity / ds test face exists.
(define erase (entity:erase dsmodel1))
;; erase
; Render the loads and constraints.
(ds:set–draw–state dsmodel1 1
    (+ ds–draw–cstrns ds–draw–loads))
;; ()
; Change the shape by tracking a point constraint.
(define c1 (ds:add–pt–cstrn dsmodel1 1
    ”position” (par–pos 0.5 0.5)))
;; c1
(ds:set–pt–xyz dsmodel1 c1 0
    (position 18 18 30))
;; 8
(ds:solve dsmodel1 1 1)
;; ()
; OUTPUT Solve 1

; Change the dynamic parameters to cause overshoot
; and wobble in future changes in shape.
(ds:set–dynamics dsmodel1 1 0.05 20.0 1.0)
;; ()
; Toggle point constraint to get the shape in motion.
(ds:toggle–cstrn dsmodel1 c1)
;; 8
; Repeated ds:solve calls show the surface moving
; which overshoots and bounces back to the origin.
; A programming interface with a repeating solve and
; render loop can show this as an animation.
(ds:solve dsmodel1 1 1)
;; ()
; OUTPUT Solve 2

(ds:solve dsmodel1 1 1)
;; ()
; OUTPUT Solve 3

(ds:solve dsmodel1 1 1)
;; ()
; OUTPUT Solve 4
```

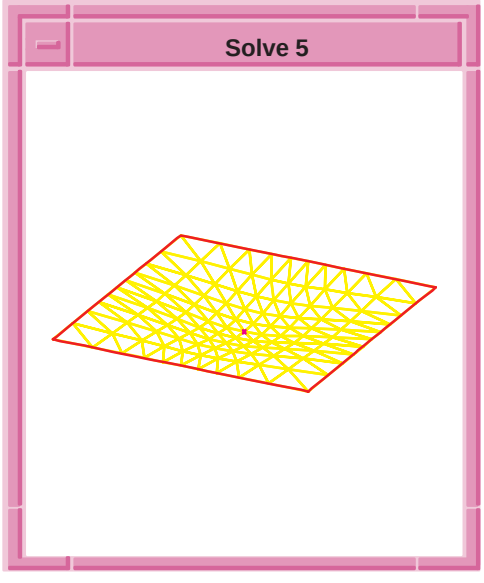
```
(ds:solve dsmodel1 1 1)
;; ()
; OUTPUT Solve 5

(ds:solve dsmodel1 1 1)
;; ()
; OUTPUT Solve 6

(ds:solve dsmodel1 1 1)
;; ()
; OUTPUT Solve 7

(ds:solve dsmodel1 1 1)
;; ()
; OUTPUT Solve 8

; Use more (ds:solve) calls to see the motion decay.
; Get the current set of dynamics parameters
(ds:get–dynamics dsmodel1 1)
;; (0.05 20 1)
```

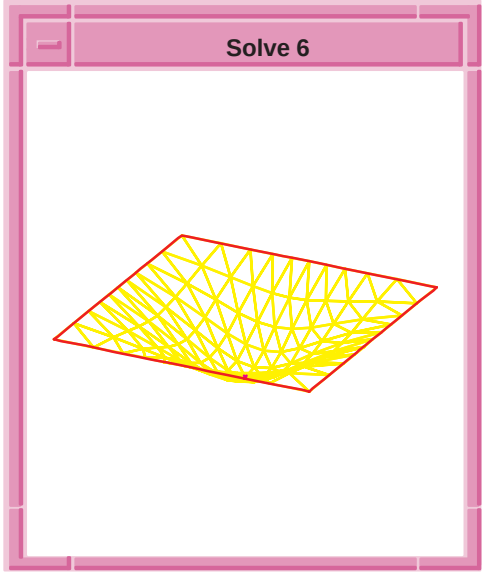





Solve 3


Solve 4


Solve 5


Solve 6

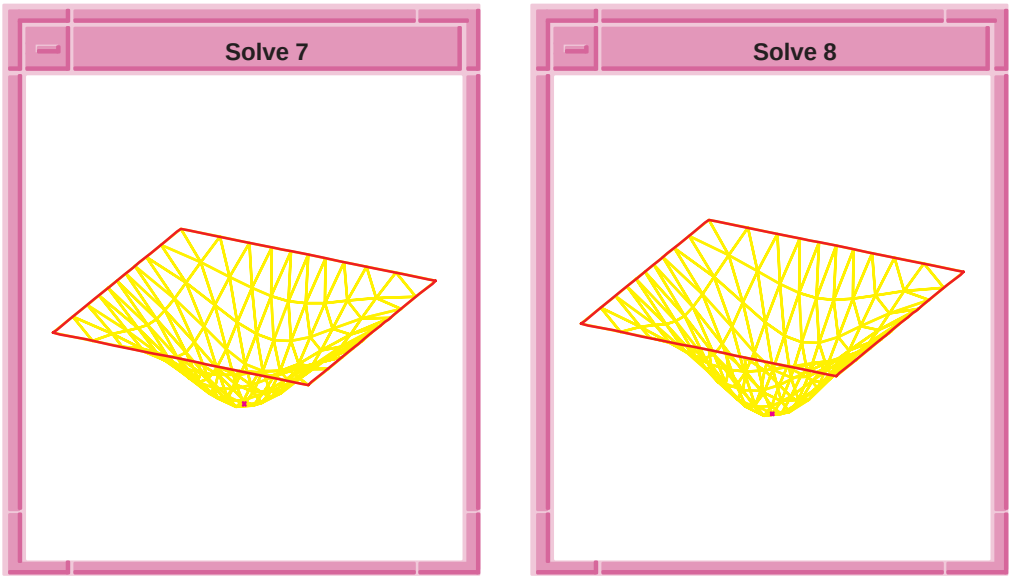


Figure 5-2. ds:get_dynamics

ds:get–entities

Scheme Extension: Deformable Surfaces

Action: This extension returns all the entities (FACE's or EDGE's) associated with a deformable modeling hierarchy.

Filename: adm/ds_scm/dsscm.cxx

APIs: api_dm_get_attrib_dm2acis, api_dm_get_hierarchy_entities

Syntax: `(ds:get–entities owner)`

Arg Types: owner entity

Returns: entity

Errors: None

Description: In ACIS deformable modeling, every deformable model corresponds to one or more ACIS ENTITIES, either FACEs or EDGEs. This query returns the ENTITY's associated with all the DS_dmods in the deformable modeling hierarchy.

owner ACIS face or edge on which the deformable model lives.

Limitations: None

Example:
```
; ds:get–entities
; No example available at this time
```

ds:get–entity

Scheme Extension: Deformable Surfaces

Action: This extension returns the entity (FACE or EDGE) associated with a deformable model.

Filename: adm/ds_scm/dsscm.cxx

APIs: api_dm_get_attrib_dm2acis

Syntax: `(ds:get–entity owner tag)`

Arg Types:		
	owner	entity
	tag	integer

Returns: entity

Errors: None

Description: In ACIS deformable modeling, every deformable model corresponds to exactly one ACIS entity, either a FACE or an EDGE. This query returns the entity associate with the underlying ACIS entity.

owner ACIS face or edge on which the deformable model lives.

tag identifier recognizes a constraint.

Limitations: None

Example:
```
; ds:get–entity
; No example at this time
```

ds:get–epsilon

Scheme Extension: Deformable Surfaces

Action: Gets the epsilon fairing parameter for a deformable model.

Filename: adm/ds_scm/dsscm.cxx

APIs: api_dm_get_attrib_dm2acis

Syntax: (**ds:set–epsilon** owner target)

Arg Types:

owner	entity
target	integer

Returns: real

Errors: None

Description: Epsilon regulates a shape fairing (energy minimization) term that is used to dampen control point oscillations in high degree splines (degree > 8).

Like the primary fairing terms, alpha and beta, epsilon should be 0 or positive. Epsilon is considered a supplement to the alpha and beta shape fairing terms, and should be relatively small compared to beta, the chief shape fairing term.

owner ACIS face or edge on which the deformable model lives.

target specifies which deformable model to use in a patch hierarchy.

Limitations: None

Example:

```
; ds:get–epsilon
; Create a block.
(define b1 (solid:block (position 5 10 15)
    (position 10 20 30)))
;; b1
; pick a face
(ray:queue 8.00781 14.6484 500 0 0 –1 1)
;; #[ray (8.00781 14.6484 500) (0 0 –1)]
(define ds–model (pick–face))
;; ds–model
(define eps (ds:get–epsilon ds–model 2))
;; eps
(print eps)
;; 0
(ds:set–epsilon ds–model 2 0.1)
;; ()
(define eps (ds:get–epsilon ds–model 2))
;; eps
(print eps)
;; 0.1
```

ds:get–gamma

Scheme Extension: Deformable Surfaces

Action: Gets the owner's deformable model's resistance to bending rate of change parameter for the deformable model.

Filename: adm/ds_scm/dsscm.cxx

APIs: api_dm_get_attrib_dm2acis

Syntax: `(ds:get–gamma owner [target=1])`

Arg Types:		
	owner	entity
	target	integer

Returns: real

Errors: None

Description: Returns the owner's deformable model's resistance to bending rate of change parameter value for the target deformable model. The gamma term is to be used in conjunction with curvature constraints and C2 seams used to connect parent and child patches together with C2 continuity. The curvature constraint affects the curvature of the deformable model only at the point at which it is applied. The gamma weighted resistance to bending changes will blend the curvature constraint effect in with the rest of the deformable modeling constraint. When gamma is zero, it has no effect on the system.

The target argument specifies which deformable model to use in a patch hierarchy. Valid values for target are:

1 = active deformable model
2 = root deformable model
–1 = active deformable model and offspring
–2 = root deformable model and offspring

Otherwise, the target is the deformable model whose tag identifier equals target.

owner ACIS face or edge on which the deformable model lives.

target specifies which deformable model to use in a patch hierarchy.

Limitations: None

Example:

```
; ds:get–gamma
; Define some helpful globals
; Build a test square face with a constraint point.
; (6x6 control points, x and y side length = 36)
(define dsmodel1 (ds:test–face 20 20 36 36 0))
;; dsmodel1
; Don’t render the face.
(define c1 (ds:add–pt–cstrn dsmodel1 1
    ”position” (par–pos .5 .5)))
;; c1
(ds:set–pt–xyz dsmodel1 c1 0
    (position 18 18 20))
;; 8
; Solve for the shape with and with default–shape.
(ds:set–gamma dsmodel1 2 0.0)
;; ()
(ds:solve dsmodel1 2 1)
;; ()
; The surface deforms.
(ds:set–gamma dsmodel1 2 40)
;; ()
(ds:get–gamma dsmodel1)
;; 40
(ds:solve dsmodel1 2 1)
;; ()
```

ds:get–icon–radius

Scheme Extension: Deformable Surfaces

Action: Gets the current icon radius being used to render tag object information.

Filename: adm/ds_scm/dsscm.cxx

APIs: api_dm_get_attrib_dm2acis

Syntax: (**ds:get–icon–radius** owner)

Arg Types:	owner	entity

Returns: real

Errors: None

Description: Returns the current icon radius size for rendering tag object information.

owner ACIS face or edge on which the deformable model lives.

Limitations: None

Example:
```
; ds:get–icon–radius
; Build a test square spline face.
; (6x6 control points, x and y side length = 36)
(define dsmodel1 (ds:test–face 6 6 36 36 0))
;; dsmodel1
; Don't display entity / ds test face exists.
(define erase (entity:erase dsmodel1))
;; erase
; Render the loads and constraints.
(ds:set–draw–state dsmodel1 1
    (+ ds–draw–cstrns ds–draw–loads))
;; ()
; Ensure the dof state is current.
(ds:solve dsmodel1 1 1)
;; ()
; Get the icon–radius state for the surface.
(ds:get–icon–radius dsmodel1)
;; 1
```

ds:get–integral–degree

Scheme Extension: Deformable Surfaces

Action: Gets the accuracy of spatial integrations executed on a deformable model.

Filename: adm/ds_scm/dsscm.cxx

APIs: api_dm_get_attrib_dm2acis

Syntax: (**ds:get–integral–degree** owner [target=1])

Arg Types:

owner	entity
target	integer

Returns: unspecified

Errors: None

Description: Gets the accuracy of the spatial integrations used for this owner's target deformable model.

The target argument specifies which deformable model to use in a patch hierarchy. Valid values for target are:

1 = active deformable model
2 = root deformable model
–1 = active deformable model and offspring
–2 = root deformable model and offspring

Otherwise, the target is the deformable model whose tag identifier equals target.

Deformable models evaluate integrals numerically when building the surface's equations of motion and the equations for constraint and load curves. Deformable surfaces are initially built with this value set to 6. Users may want to increase this value if they have found a curve constraint which is not being maintained adequately. Increasing this number beyond ten will tend to have little results on the system but begin to slow the system down considerably.

owner ACIS face or edge on which the deformable model lives.

target specifies which deformable model to use in a patch hierarchy.

Limitations: None

Example:

```
; ds:get–integral–degree
; Build a test square face with some tag objects
; (6x6 control points, x and y side length = 36)
(define dsmodel1 (ds:test–face 6 6 36 36 0))
;; dsmodel1
; Don't display entity / ds test face exists.
(define erase (entity:erase dsmodel1))
;; erase
; Change the accuracy of gauss integration.
(ds:get–integral–degree dsmodel1 1)
;; 10
```

ds:get–interior–state

Scheme Extension: Deformable Surfaces

Action: Gets the interior state value for a target deformable model.

Filename: adm/ds_scm/dsscm.cxx

APIs: api_dm_get_attrib_dm2acis

Syntax: `(ds:get–interior–state owner [target=1])`

Arg Types:	owner	entity
	target	integer

Returns: integer

Errors: None

Description: Gets the interior state value for the target deformable model.

When the interior_state = 0, the Deformable model is allowed to bend with C0 discontinuity between elements. For Bsplines and NURBs, C0 discontinuity within a surface can be achieved by increasing the knot count at an internal knot boundary.

When interior_state = 1, the Deformable model prohibits C0 bending between elements by adding C1 internal tangent constraints between any elements whose internal representation will allow a C0 bend. For Bsplines and NURBs, this means that the deformation will be at least C1 everywhere, even if the underlying representation has multiple knots that would normally allow a C0 internal bend.

The target argument specifies which deformable model to use in a patch hierarchy. Valid values for target are:

1 = active deformable model
2 = root deformable model
–1 = active deformable model and offspring
–2 = root deformable model and offspring

Otherwise, the target is the deformable model whose tag identifier equals target.

owner ACIS face or edge on which the deformable model lives.

target specifies which deformable model to use in a patch hierarchy.

Limitations: None

Example:

```
; ds:get–interior–state
; Build a test square face with some tag objects
; (6x6 control points, x and y side length = 36)
(define dsmodel1 (ds:test–face 6 6 36 36 0))
;; dsmodel1
; Don't display entity / ds test face exists.
(define erase (entity:erase dsmodel1))
;; erase
; Render the control–points and tag objects.
(ds:set–draw–state dsmodel1 1
    (+ ds–draw–cpts ds–draw–cstrns ds–draw–loads))
;; ()
(ds:solve dsmodel1 1 1)
;; ()
(ds:get–interior–state dsmodel1)
;; 1
; The system returns the default interior state
; value.
(ds:set–interior–state dsmodel1 1 0)
;; ()
(ds:get–interior–state dsmodel1)
;; 0
; The system returns assigned interior state value.
```

ds:get–load–gain

Scheme Extension: Deformable Surfaces

Action: Gets the current gain value of a load tag object.

Filename: adm/ds_scm/dsscm.cxx

APIs: api_dm_get_attrib_dm2acis

Syntax: (**ds:get–load–gain** owner tag)

Arg Types:

owner	entity
tag	integer

Returns: real

Errors: None

Description: Gets the current gain of one tag object in the face's deformable model.

The tag object is identified by the tag value. When the tag identifies a tag object, the deformable model in the patch hierarchy which contains the tag object is made the active deformable model.

owner ACIS face or edge on which the deformable model lives.

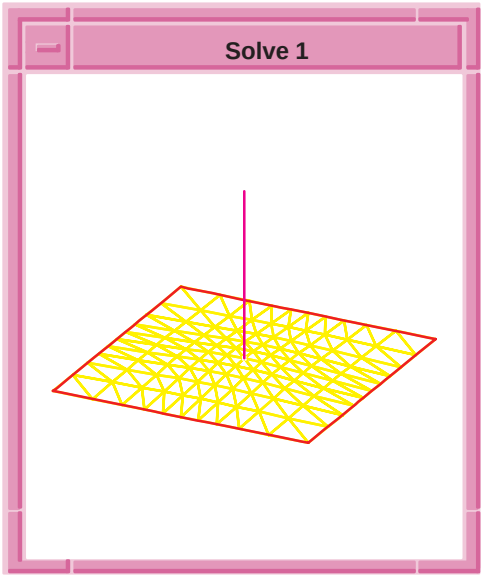
tag identifier recognizes a constraint.

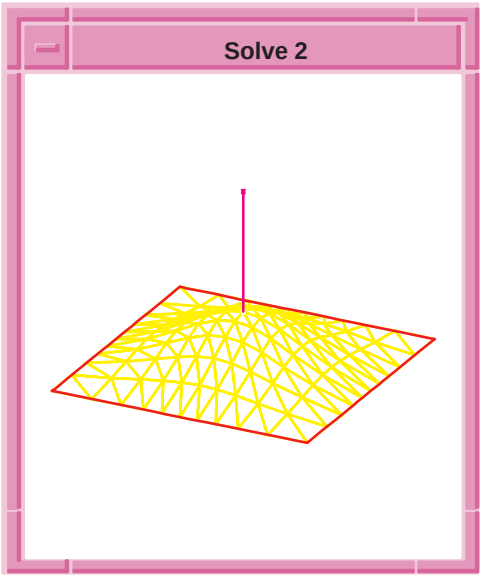
Limitations: None

Example:

```
; ds:get–load–gain
; Build a test square face with some tag objects
; (6x6 control points, x and y side length = 36)
(define dsmodel1 (ds:test–face 6 6 36 36 0))
;; dsmodel1
; Don’t display entity / ds test face exists
(define erase (entity:erase dsmodel1))
;; erase
; Render the loads and constraints.
(ds:set–draw–state dsmodel1 1
    (+ ds–draw–cstrns ds–draw–loads))
;; ()
; Add a spring to deform the surface.
(define c1 (ds:add–spring dsmodel1 1
    (par–pos 0.5 0.5) (position 18 18 25) 10))
;; c1
(ds:solve dsmodel1 1 1)
;; ()
; OUTPUT Solve 1

; The surface should just barely be deformed.
; Increasing the spring gain reduces the distance
; between the spring end points.
(ds:set–load–gain dsmodel1 c1 100 #t)
;; 4
(ds:solve dsmodel1 1 1)
;; ()
; OUTPUT Solve 2

(ds:set–load–gain dsmodel1 c1 100 #t)
;; 4
(ds:solve dsmodel1 1 1)
;; ()
; OUTPUT Solve 3

(ds:set–load–gain dsmodel1 c1 100 #t)
;; 4
(ds:solve dsmodel1 1 1)
;; ()
; OUTPUT Solve 4
```

```
; The spring displaces the surface by larger
; and larger amounts.
; Check the final load value for the spring.
(ds:get-load-gain dsmodel1 c1)
;; 310
```





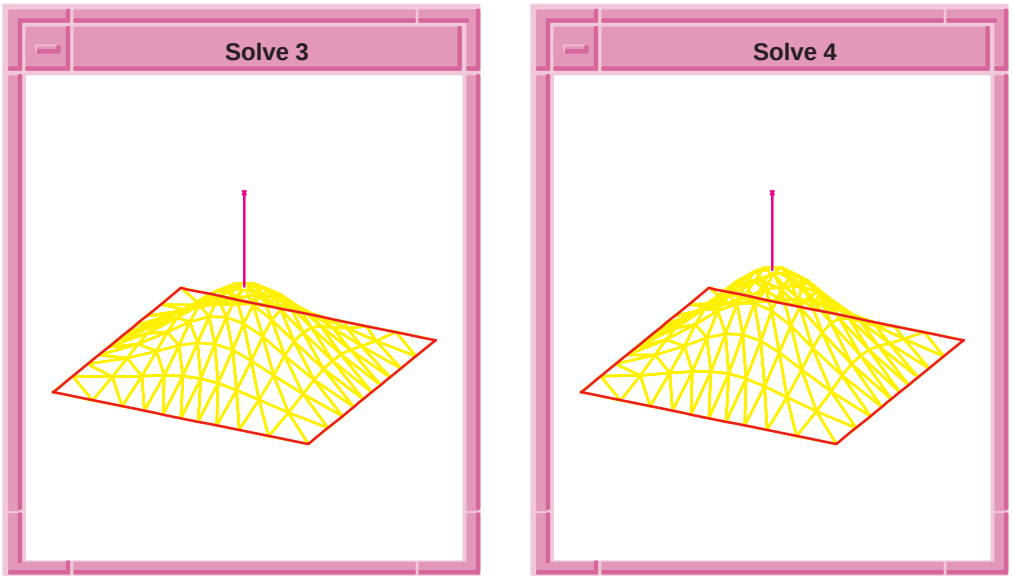


Figure 5-3. ds:get–load–gain

ds:get–minimal–corners

Scheme Extension: Deformable Surfaces

Action: This extension returns the minimal corners for an area load.

Filename: adm/ds_scm/dsscm.cxx

APIs: api_dm_get_attrib_dm2acis, api_dm_set_array_size

Syntax: `(ds:get-minimal-corners owner tag)`

Arg Types:

owner	entity
tag	integer

Returns: real ...

Errors: None

Description: An area load restricts behavior on a subdomain of the DS_dmod. This function returns the minimal corner points characterizing this subdomain for an area load.

owner ACIS face or edge on which the deformable model lives.

tag identifier recognizes a constraint.

Limitations: None

Example:
```
; ds:get–minimal–corners
; No example available at this time
```

ds:get–parent–tag

Scheme Extension: Deformable Surfaces, DML Tags

Action: Returns the tag number of the target deformable model's parent or –1 for none.

Filename: adm/ds_scm/dsscm.cxx

APIs: api_dm_get_attrib_dm2acis

Syntax: `(`**`ds:get–parent–tag`** `owner [target=1])`

Arg Types:

owner	entity
target	integer

Returns: integer

Errors: None

Description: Returns the tag identifier for the target dormable model's parent or –1 for none.

The target argument specifies which deformable model to use in a patch hierarchy. Valid values for target are:

1 = active deformable model
2 = root deformable model

Otherwise, the target is the deformable model whose tag identifier equals target.

owner ACIS face or edge on which the deformable model lives.

target specifies which deformable model to use in a patch hierarchy.

Limitations: None

Example:

```
; ds:get–parent–tag
; Build a test square spline face.
; (6x6 control points, x and y side length = 36)
(define dsmodel1 (ds:test–face 6 6 36 36 0))
;; dsmodel1
; Don't display entity / ds test face exists.
(define erase (entity:erase dsmodel1))
;; erase
; Render the loads and constraints.
(ds:set–draw–state dsmodel1 1
    (+ ds–draw–cstrns ds–draw–loads))
;; ()
; Add a rectangular patch to the parent shape.
; The new patch becomes the active shape.
(define patch1 (ds:add–patch dsmodel1 1 1
    (par–pos 0.3 0.3) (par–pos 0.5 0.5)
    (par–pos 0.7 0.7)3))
;; patch1
; Exercise ds:get–parent–tag.
(ds:get–parent–tag dsmodel1 patch1)
;; 2
```

ds:get–pt–uv

Scheme Extension: Deformable Surfaces

Action: Gets the parametric position within a deformable surface of a constraint point, a pressure point, or a spring load.

Filename: adm/ds_scm/dsscm.cxx

APIs: api_dm_get_attrib_dm2acis

Syntax: (**ds:get–pt–uv** owner tag)

Arg Types:

owner	entity
tag	integer

Returns: par–pos

Errors: None

Description: Gets the surface point location of a constraint point, a pressure point, or a spring load in the face's deformable model. tag identifies the tag object to query. The *u* and *v* values in uv_pos are scaled to range from 0.0 to 1.0.

When the tag identifies a tag object, the deformable model in the patch hierarchy which contains the tag object is made the active deformable model.

owner ACIS face or edge on which the deformable model lives.

tag identifier recognizes a constraint.

Limitations: None

Example:

```
; ds:get–pt–uv
; Build a test square face with some tag objects
; (6x6 control points, x and y side length = 36)
(define dsmodel1 (ds:test–face 6 6 36 36 0))
;; dsmodel1
; Don't display entity / ds test face exists.
(define erase (entity:erase dsmodel1))
;; erase
; Render the loads and constraints.
(ds:set–draw–state dsmodel1 1
    (+ ds–draw–cstrns ds–draw–loads))
;; ()
; Add center point constraints.
(define cc1 (ds:add–pt–cstrn dsmodel1
    1 "position" (par–pos 0.5 0.5)))
;; cc1
(ds:solve dsmodel1 1 1)
;; ()
; Get the surface location of the pt constraint.
(ds:get–pt–uv dsmodel1 cc1)
;; #[par–pos 0.5 0.5]
```

ds:get–pt–xyz

Scheme Extension: Deformable Surfaces

Action: Gets the position within *xyz* space of a control point, a constraint point, or a spring load.

Filename: adm/ds_scm/dsscm.cxx

APIs: api_dm_get_attrib_dm2acis

Syntax: (**ds:get–pt–xyz** owner tag [point–index=0])

Arg Types:		
	owner	entity
	tag	integer
	point–index	integer

Returns: position

Errors: None

Description: Returns the three dimensional space point position for a control point, a constraint point, or a spring load within the face's deformable model.

The point–index identifies which image point object to query when the given tag number identifies a tag object with more than one image point. tag identifies the tag object to query.

When tag does not identify an object with a distinct image point, unspecified is returned. When the tag identifies a tag object, the deformable model in the patch hierarchy which contains the tag object is made the active deformable model.

Point constraints:

point–index = 0	the base–point
point–index = 2	the curve tangent end–point
point–index = 2	the surface tang1 end–point
point–index = 3	the surface tang2 end–point
point–index = 4	the normal vector end–point
point–index = 5	the curve curvature end–point
point–index = 5	the surface curv1 end–point
point–index = 6	the surface curv2 end–point
point–index = 7	the curve binormal end–point

For a curve constraint, the image point equals the last point used to pick the curve.

For distributed pressure, the image point is unspecified

For point pressure, the image point is the image point of the point pressure.

For a spring load, the image point is the free point location of the spring.

For a spring set, the image point is the free point of the pt–index spring of the set.

For a curve load, the image point is unspecified

For a dynamic load, the image point is unspecified.

owner ACIS face or edge on which the deformable model lives.

point–index identifies which image point object to query.

tag identifies the tag object to query.

Limitations: None

Example:

```
; ds:get–pt–xyz
; Build a test square face with some tag objects
; (6x6 control points, x and y side length = 36)
(define dsmodel1 (ds:test–face 6 6 36 36 0))
;; dsmodel1
; Don’t display entity / ds test face exists.
(define erase (entity:erase dsmodel1))
;; erase
; Render the loads and constraints.
(ds:set–draw–state dsmodel1 1
    (+ ds–draw–cstrns ds–draw–loads))
;; ()
; Add a point constraint.
(define cc1 (ds:add–pt–cstrn dsmodel1
    1 ”position” (par–pos 0.5 0.5)))
;; constructed new attrib_dm2acis
;; cc1
(ds:solve dsmodel1 1 1)
; Track the constraint point with the mouse.
(ui:info–dialog
    ”click with mouse to move the pt–cstrn :”)
;; #t
(ds:set–pt–xyz dsmodel1 cc1 0 2 (read–event))
;; 8
(ds:solve dsmodel1 1 1)
;; ()
; Constraint point moves in the z direction and
; the Xsolve call makes the surface track the point.
; Find the 3–space location of the constraint point.
(ds:get–pt–xyz dsmodel1 cc1 0)
;: #[position 18.0 18.0 16.785]
; The z number varies depending on where you clicked.
```

ds:get–shape–degree

Scheme Extension: Deformable Surfaces

Action: Gets the degree polynomial values for a deformable model shape representation.

Filename: adm/ds_scm/dsscm.cxx

APIs: api_dm_get_attrib_dm2acis

Syntax: `(ds:get–shape–degree owner [target=1])`

Arg Types:

owner	entity
target	integer

Returns: pair

Errors: None

Description: For deformable surfaces, this returns *u* and *v* direction polynomial degree values for the owner's target deformable model basis functions. For deformable curves, this returns the one polynomial degree value for the owner's target deformable model basis function.

The target argument specifies which deformable model to use in a patch hierarchy. Valid values for target are:

1 = active deformable model
2 = root deformable model
–1 = active deformable model and offspring
–2 = root deformable model and offspring

Otherwise, the target is the deformable model whose tag identifier equals target.

(u_basis_degree, v_basis_degree) . . . for surfaces
(u_basis_degree) . . . . . for curves

owner ACIS face or edge on which the deformable model lives.

target specifies which deformable model to use in a patch hierarchy.

Limitations: None

Example:

```
; ds:get–shape–degree
; Add to a square test face a parabolic crv–load
; and use it.
; Build a test square spline face.
; (6x6 control points, x and y side length = 36)
(define dsmodel1 (ds:test–face 6 6 36 36 0))
;; dsmodel1
; Don't display entity / ds test face exists
(define erase (entity:erase dsmodel1))
;; erase
; Render the loads and constraints.
(ds:set–draw–state dsmodel1 1
    (+ ds–draw–cstrns ds–draw–loads))
;; ()
(ds:get–shape–degree dsmodel1 1)
;; (3 3)
```

ds:get–shape–dofs

Scheme Extension: Deformable Surfaces

Action: Gets arrays of the dof, default_dof, and weight values which define the current shape of deformable NURBs and NUBs.

Filename: adm/ds_scm/dsscm.cxx

APIs: api_dm_get_attrib_dm2acis

Syntax: (**ds:get–shape–dofs** owner [target=1])

Arg Types:

owner	entity
target	integer

Returns: (real ...)

Errors: None

Description: For NURB- and NUB-based deformable surfaces this extension returns the dof, default_dof, and weight arrays that define the target's current shape.

```
Returns
    (tot_dof_count u_dof_count v_dof_count
    (dof0(x y z) dof1(x y z) ...)
    (default_dof0(x y z) default_dof1(x y z) ...)
    (weight0 weight1 ...) for NURBS and
    ((dof0x y z dof1x y z ...)
    (default_dof0x y z default_dof1x y z ...)
. .. for B–splines.
```

where tot_dof_count is the total number of degrees of freedom used to define the deformable model's shape, and u_dof_count and v_dof_count are the number of dofs in the *u* and *v* directions.

The variables dofi are the dof values (a control point location) and the variables default_dofi are the default locations for those dofs. weighti are the weights associated with each dof for a NURB shape. Non–NURB shapes do not include a weight list in the return.

The target argument specifies which deformable model to use in a patch hierarchy. Valid values for target are:

1 = active deformable model
2 = root deformable model
–1 = active deformable model and offspring
–2 = root deformable model and offspring

Otherwise, the target is the deformable model whose tag identifier equals target.

owner ACIS face or edge on which the deformable model lives.

target specifies which deformable model to use in a patch hierarchy.

Limitations: None

Example:

```
; ds:get–shape–dofs
; Define some helpful globals
; Build a test square spline face.
; (6x6 control points, x and y side length = 36)
(define dsmodel1 (ds:test–face 6 6 36 36 0))
;; dsmodel1
(define erase (entity:erase dsmodel1))
;; erase
; Don't render the face.
; Get the basis degree values.
(ds:get–shape–dofs dsmodel1 1)
;; (36 6 6 (0 0 1 0 7.2 1 0 14.4 1 0 21.6 1 0 28.8 1
;; 0 36 1 7.2 0 1 7.2 7.2 1 7.2 14.4 1 7.2 21.6 1
;; 7.2 28.8 1 7.2 36 1 14.4 0 1 14.4 7.2 1 14.4 14.4
;; 1 14.4 21.6 1 14.4 28.8 1 14.4 36 1 21.6 0 1 21.6
;; 7.2 1 21.6 14.4 1 21.6 21.6 1 21.6 28.8 1 21.6 36
;; 1 28.8 0 1 28.8 7.2 1 28.8 14.4 1 28.8 21.6 1
;; 28.8 28.8 1 28.8 36 1 36 0 1 36 7.2 1 36 14.4 1
;; 36 ...) (0 0 1 0 7.2 1 0 14.4 1 0 21.6 1 0 28.8 1
;; 0 36 1 7.2 0 1 7.2 7.2 1 7.2 14.4 1 7.2 21.6 1
;; 7.2 28.8 1 7.2 36 1 14.4 0 1 14.4 7.2 1 14.4 14.4
;; 1 14.4 21.6 1 14.4 28.8 1 14.4 36 1 21.6 0 1 21.6
;; 7.2 1 21.6 14.4 1 21.6 21.6 1 21.6 28.8 1 21.6 36
;; 1 28.8 0 1 28.8 7.2 1 28.8 14.4 1 28.8 21.6 1
;; 28.8 28.8 1 28.8 36 1 36 0 1 36 7.2 1 36 14.4 1
;; 36 ...))
```

ds:get–shape–knots

Scheme Extension: Deformable Surfaces

Action: Gets arrays of the *u* and *v* direction knot values which act as element boundaries for deformable NURBs and NUBs.

Filename: adm/ds_scm/dsscm.cxx

APIs: api_dm_get_attrib_dm2acis

Syntax: `(ds:get-shape-knots owner [target=1])`

Arg Types:

owner	entity
target	integer

Returns: (real ...)

Errors: None

Description: For NURB and NUB based deformable surfaces returns *u* and *v* knot vectors for the owner's target model B–spline basis. For deformable curves returns the *u* knot vector for the owner's target deformable model B–spline basis.

```
Returns
. .. ((u0 u1 ...) (v0 v1 ...)
. .. (ui0 ui1 ...) (vi0 vi1 ...) for surfaces and
. .. ((u0 u1 ...) (v0 v1 ...) for curves.
```

The target argument specifies which deformable model to use in a patch hierarchy. Valid values for target are:

1 = active deformable model
2 = root deformable model
–1 = active deformable model and offspring
–2 = root deformable model and offspring

Otherwise, the target is the deformable model whose tag identifier equals target.

The multiplicity of each knot is communicated through the knot_index array. Each knot, starting with the lowest *u* value knot, is assigned a knot index starting at 0. The knot index array records the highest index value for all knots at each distinct knot location. For example, a common B–spline is one with multiple knots on the end points and single knots in the interior. A map of the knot index values might look like,

```
+------+------+-------+------+
0       3       4       5       6
1                               7
2                               8
```

This knot array corresponds to a degree=3 B–spline with 7 control points and 4 spans. The B–spline interpolates the first and last control point positions due to the multiple end knots.

owner ACIS face or edge on which the deformable model lives.

target specifies which deformable model to use in a patch hierarchy.

Limitations: None

Example:
```
; ds:get–shape–knots
; Define some helpful globals.
;; ds–model
; Build a test square spline face.
; (6x6 control points, x and y side length = 36)
(define dsmodel1 (ds:test–face 6 6 36 36 0))
;; dsmodel1
(define erase (entity:erase dsmodel1))
;; erase
; Don't render the face.
; Get the basis degree values.
(ds:get–shape–knots dsmodel1 1)
;; ((0 0.333333333333333 0.666666666666667 1)
;; (0 0.333333333333333 0.666666666666667 1)
;; (2 3 4 7) (2 3 4 7))
```

ds:get–sibling–tag

Scheme Extension: Deformable Surfaces, DML Tags

Action: Returns the tag number of the target deformable model's sibling or –1 for none.

Filename: adm/ds_scm/dsscm.cxx

APIs: api_dm_get_attrib_dm2acis

Syntax: (**ds:get–sibling–tag** owner [target=1])

Arg Types:		
	owner	entity
	target	integer

Returns: integer

Errors: None

Description: Returns the tag identifier for the target deformable model's sibling or –1 for none.

The target argument specifies which deformable model to use in a patch hierarchy. Valid values for target are:

1 = active deformable model
2 = root deformable model

Otherwise, the target is the deformable model whose tag identifier equals target.

owner ACIS face or edge on which the deformable model lives.

target specifies which deformable model to use in a patch hierarchy.

Limitations: None

Example:

```
; ds:get–sibling–tag
; Build a test square spline face.
; (6x6 control points, x and y side length = 36)
(define dsmodel1 (ds:test–face 6 6 36 36 0))
;; dsmodel1
; Erase the display of this entity / ds test face.
;   exists
(define erase (entity:erase dsmodel1))
;; erase
; Render the loads and constraints.
(ds:set–draw–state dsmodel1 1
    (+ ds–draw–cstrns ds–draw–loads))
;; ()
; Add two rectangular patches to the sibling shape.
; The new patch becomes the active shape.
(define patch1 (ds:add–patch dsmodel1 2 1
    (par–pos 0.3 0.3) (par–pos 0.5 0.5)
    (par–pos 0.0 0.0)3))
;; patch1
(define patch2 (ds:add–patch dsmodel1 2 1
    (par–pos 0.7 0.7) (par–pos 0.9 0.9)
    (par–pos 0.0 0.0) 3))
;; patch2
; Get the tag identifier for this model.
(ds:get–sibling–tag dsmodel1 patch1)
;; 12
```

ds:get–spring–length

Scheme Extension: Deformable Surfaces

Action: Sets the gain of a load tag object.

Filename: adm/ds_scm/dsscm.cxx

APIs: api_dm_get_attrib_dm2acis

Syntax: `(`**`ds:get–spring–length`** `owner target=1 gain=100 [inc])`

Arg Types:	owner	entity
	target	integer
	gain	real
	inc	boolean

Returns: integer

Errors: None

Description: Sets or increments the gain on one or more load target objects in the owner's deformable model.

The target argument specifies which deformable model to use in a patch hierarchy. Valid values for target are:

1 = active deformable model
2 = root deformable model
–1 = active deformable model and offspring
–2 = root deformable model and offspring

Otherwise, the target is the deformable model whose tag identifier equals target.

When the tag type is –2, all point pressure load gains are modified. When the tag type is–3, all spring load gains are modified. When the tag type is a positive integer, only the object with that tag identifier is modified.

When the inc argument is omitted, the modified load gains are set to the input gain value. When the inc argument is included, the modified load gains are incremented by the input gain value.

owner ACIS face or edge on which the deformable model lives.

target specifies which deformable model to use in a patch hierarchy.

gain is a measure of how strongly the load pulls the deformable model to its target.

inc is used to control modified load gains.

Limitations: None

Example:

```
; ds:get–spring–length
; Add to a square test face a parabolic crv–load
; and use it.
; Build a test square spline face.
; (6x6 control points, x and y side length = 36)
(define dsmodel1 (ds:test–face 6 6 36 36 0))
;; dsmodel1
; Don't display entity / ds test face exists.
(define erase (entity:erase dsmodel1))
;; erase
; Render the loads and constraints.
(ds:set–draw–state dsmodel1 1
    (+ ds–draw–cstrns ds–draw–loads))
;; ()
(define cc1 (ds:add–spring dsmodel1
    1 (par–pos 0.5 0.5) (position 18 18 25) 10))
;; cc1
; Compute a new deformable model position.
(ds:solve dsmodel1 1 1)
;; ()
; Increasing the spring gain reduces the distance
; between the spring end points.
(ds:get–spring–length dsmodel1 cc1)
;; 23.0688847300202
(ds:solve dsmodel1 1 1)
;; ()
```

ds:get–tag–param–max

Scheme Extension: Deformable Surfaces

Action: This extension returns an upper bound for the tag object parameterization.

Filename: adm/ds_scm/dsscm.cxx

APIs: api_dm_get_attrib_dm2acis

Syntax: (**ds:get–tag–param–max** owner tag)

Arg Types:

owner	entity
tag	integer

Returns: (real ...)

Errors: None

Description: Tag objects are usually parameterized. For example, a curve is parameterized by a single continuous parameter and a spring-set is parameterized by a single discrete parameter. This extension returns an upper bound for the tag object parameterization.

owner ACIS face or edge on which the deformable model lives.

tag identifier recognizes a constraint.

Limitations: None

Example:

```
; ds:get–tag–param–max
; build a sheet–body
(define my_sheet (sheet:face
    (face:plane (position 0 –5 0) 10 10)))
;; my_sheet
; get the face
(define ds–model (list–ref
    (entity:faces my_sheet) 0))
;; ds–model
; get a nice view
(iso)
;; #[view 852730]
(zoom–all)
;; #[view 852730]
; create a curve load
(define crv–load (ds:add–str–load ds–model 2
    (par–pos .1 .1) (par–pos .3 .4) 1000))
;; crv–load
; query for param max
(ds:get–tag–param–max ds–model crv–load)
;; (1)
```

ds:get–tag–param–min

Scheme Extension: Deformable Surfaces

Action: This extension returns a lower bound for the tag object parameterization.

Filename: adm/ds_scm/dsscm.cxx

APIs: api_dm_get_attrib_dm2acis

Syntax: `(`**`ds:get–tag–param–min`** `owner tag)`

Arg Types:		
	owner	entity
	tag	integer

Returns: (real ...)

Errors: None

Description: Tag objects are usually parameterized. For example, a curve is parameterized by a single continuous parameter and a spring-set is parameterized by a single discrete parameter. This method returns a lower bound for the tag object parameterization.

owner ACIS face or edge on which the deformable model lives.

tag identifier recognizes a constraint.

Limitations: None

Example:

```
; ds:get–tag–param–min
; build a sheet–body
(define my_sheet (sheet:face
    (face:plane (position 0 –5 0) 10 10)))
;; my_sheet
; get the face
(define ds–model (list–ref
    (entity:faces my_sheet) 0))
;; ds–model
; get a nice view
(iso)
;; #[view 852730]
(zoom–all)
;; #[view 852730]
; create a curve load
(define crv–load (ds:add–str–load ds–model 2
    (par–pos .1 .1) (par–pos .3 .4) 1000))
;; crv–load
; query for param min
(ds:get–tag–param–min ds–model crv–load)
;; (0)
```

ds:get–tag–patch

Scheme Extension: Deformable Surfaces

Action: Gets the tag identifier number of the patch which contains the input tag object.

Filename: adm/ds_scm/dsscm.cxx

APIs: api_dm_get_attrib_dm2acis

Syntax: `(ds:get-tag-patch owner tag)`

Arg Types:

owner	entity
tag	integer

Returns: integer

Errors: None

Description: Returns the tag identifier number for the patch within the deformable model patch hierarchy associated with the input owner that contains the tag object. When the input tag value does not correspond to a tag object in the hierarchy, –1 is returned.

Making the query on an owner without deformable model causes a deformable model to be added to the entity and a tag value is returned.

owner ACIS face or edge on which the deformable model lives.

tag identifier recognizes a constraint.

Limitations: None

Example:

```
; ds:get–tag–patch
; Add to a square test face a parabolic crv–load
; and use it.
; Build a test square spline face.
; (6x6 control points, x and y side length = 36)
(define dsmodel1 (ds:test–face 6 6 36 36 0))
;; dsmodel1
; Don't display entity / ds test face exists.
(define erase (entity:erase dsmodel1))
;; erase
; Render the loads and constraints.
(ds:set–draw–state dsmodel1 1
    (+ ds–draw–cstrns ds–draw–loads))
;; ()
; Add a pt–cstrn at the center of the parent and
; track it.
(define cc1 (ds:add–pt–cstrn dsmodel1
    1 "position" (par–pos 0.5 0.5)))
;; cc1
(ds:set–pt–xyz dsmodel1 cc1 0
    (position 18 18 10))
;; 8
; Compute a new deformable model position.
(ds:solve dsmodel1 1 1)
;; ()
; Add a patch to the parent shape.
; The new patch becomes the active shape.
(ds:add–patch dsmodel1 2 1 (par–pos 0.3 0.3)
    (par–pos 0.5 0.5) (par–pos 0.0 0.0) 3))
;; 8
; Add and track a pt–cstrn on the patch.
(define cc2 (ds:add–pt–cstrn dsmodel1
    1 "position" (par–pos 0.5 0.5)))
;; cc2
(ds:set–pt–xyz dsmodel1 cc2 0
    (position 18 18 36))
;; 8
(ds:solve dsmodel1 1 1)
;; ()
; Now find the tag value of the patch that owns
; the cc2 constraint.
(ds:get–tag–patch dsmodel1 cc2)
;; 8
```

ds:get–tag–summary

Scheme Extension: Deformable Surfaces

Action: Pretty prints and returns a list of the current tag identifiers and the associated tag types for all tag objects in a deformable surface.

Filename: adm/ds_scm/dsscm.cxx

APIs: api_dm_get_attrib_dm2acis

Syntax: `(ds:get–tag–summary owner [target=1 print=1])`

Arg Types:

owner	entity
target	integer
print	integer

Returns: ((integer . integer) ...)

Errors: None

Description: Returns a list of all the tag objects currently contained within the target deformable model. Two pieces of information are returned for every contained tag object: the tag object's tag number and the tag object's type. Both pieces of information are returned as integers.

The first 8 entries of the returned list are always used to describe the target deformable model, and its parent, its sibling, and its child. Whenever the deformable model does not have a parent, sibling, or child, the associated tag_number is set to –1, and the associated tag type is set to 0.

The returned list is organized as:

```
(target Deformable Model tag_number,
    type, target's parent Deformable Model
    tag_number, type target's sibling Deformable
    Model tag_number, type, target's child Deformable
    Model tag_number, type, contained tag_number,
    type, contained tag_number, type, ... )
```

The returned list length is 2 times the number of contained tag objects.

By default, the function also pretty prints the tag list to the Scheme window.

The target argument specifies which deformable model to use in a patch hierarchy. Valid values for target are:

1 = active deformable model
2 = root deformable model
–1 = active deformable model and offspring
–2 = root deformable model and offspring

Otherwise, the target is the deformable model whose tag identifier equals target.

The print argument controls output to the Scheme window. When print equals 1 a table of the the tag values and their tag types is output. When print equals 0 no table is printed, only the (tag type ...) list is returned.

The list of returned type values includes:

–1 = No such tag object
1 = Spline Control Point
2 = Point Pressure Load
3 = Distributed Pressure Load
4 = Spring Load
5 = Spring Set Load
6 = Load Curve
7 = Dynamic Load
8 = Point Constraint (not a seam)
9 = Seam Point Constraint
10 = Curve Constraint (not a seam)
11 = Seam Curve Constraint
12 = Deformable Surface
13 = Deformable Curve
14 = Vector Load
15 = Attractor Load
16 = Multi–surface link constraint
17 = Area constraint
18 = Link Load
19 = Tracking Position Deformable Curve
20 = Tracking Tangent Deformable Curve
21 = Tracking Curvature Deformable Curve

owner ACIS face or edge on which the deformable model lives.

target specifies which deformable model to use in a patch hierarchy.

print controls output to the Scheme window.

Limitations: None

Example:

```
; ds:get–tag–summary
; Build a test square spline face.
; (6x6 control points, x and y side length = 36)
(define dsmodel1 (ds:test–face 6 6 36 36 0))
;; dsmodel1
; Don't display entity / ds test face exists.
(define erase (entity:erase dsmodel1))
;; erase
```

```
; Render the loads and constraints.
(ds:set–draw–state dsmodel1 1
    (+ ds–draw–cstrns ds–draw–loads))
;; ()
; Add a spring.
(define c1 (ds:add–spring dsmodel1 1
    (par–pos 0.5 0.5) (position 18 18 15) 100))
;; c1

; Get the list of current tag identifiers and types.
(ds:get–tag–summary dsmodel1 1)
; tgt_dmod = 2, Deformable Surface
; parent = NULL
; sibling = NULL
; child = NULL
; id type
; ––– ––––
; 3 Curve Constraint
; 4 Curve Constraint
; 5 Curve Constraint
; 6 Curve Constraint
; 7 Point Spring Load
;; (2 12 –1 0 –1 0 –1 0 3 6 4 6 5 6 6 6 7 4)
```

ds:get–tag–type

Scheme Extension: Deformable Surfaces

Action: Gets the tag object's type ID as an integer.

Filename: adm/ds_scm/dsscm.cxx

APIs: api_dm_get_attrib_dm2acis

Syntax: (**ds:get–tag–type** owner tag)

Arg Types:		
	owner	entity
	tag	integer

Returns: integer

Errors: None

Description: Returns the type of the tag associated with the input face:

The list of returned type values includes:

–1 = No such tag object
1 = Spline Control Point
2 = Point Pressure Load
3 = Distributed Pressure Load
4 = Spring Load
5 = Spring Set Load
6 = Load Curve
7 = Dynamic Load
8 = Point Constraint (not a seam)
9 = Seam Point Constraint
10 = Curve Constraint (not a seam)
11 = Seam Curve Constraint
12 = Deformable Surface
13 = Deformable Curve
14 = Vector Load
15 = Attractor Load
16 = Multi–surface link constraint
17 = Area constraint
18 = Link Load
19 = Tracking Position Deformable Curve
20 = Tracking Tangent Deformable Curve
21 = Tracking Curvature Deformable Curve

owner ACIS face or edge on which the deformable model lives.

tag identifier recognizes a constraint.

Limitations: None

Example:

```
; ds:get–tag–type
; Get the tag type of a spring load.
; Build a test square spline face.
; (6x6 control points, x and y side length = 36)
(define dsmodel1 (ds:test–face 6 6 36 36 0))
;; dsmodel1
; Don't display entity / ds test face exists.
(define erase (entity:erase dsmodel1))
;; erase
; Render the loads and constraints.
(ds:set–draw–state dsmodel1 1
    (+ ds–draw–cstrns ds–draw–loads))
;; ()
; Add a spring and solve for the new shape.
(define c1 (ds:add–spring dsmodel1 1
    (par–pos 0.5 0.5) (position 18 18 15) 100))
;; c1
; Get the tag type of the spring load.
(ds:get–tag–type dsmodel1 c1)
;; 4
```

ds:get–tags

Scheme Extension: Deformable Surfaces

Action: Returns the tags in a deformable modeling hierarchy.

Filename: adm/ds_scm/dsscm.cxx

APIs: api_dm_get_attrib_dm2acis

Syntax: (**ds:get–tags** owner)

Arg Types: owner entity

Returns: (integer ...)

Errors: None

Description: Returns an integer list of tags corresponding to all tag objects in the deformable modeling hierarchy.

owner ACIS face or edge on which the deformable model lives.

Limitations: None

Example:

```
; ds:get–tags
; Get the tags in the hierarchy.
; Build a test square spline face.
; (6x6 control points, x and y side length = 36)
(define dsmodel1 (ds:test–face 6 6 36 36 0))
;; dsmodel1
; Don't display entity / ds test face exists.
(define erase (entity:erase dsmodel1))
;; erase
; Render the loads and constraints.
(ds:set–draw–state dsmodel1 1
    (+ ds–draw–cstrns ds–draw–loads))
;; ()
; Add a spring and solve for the new shape.
(define c1 (ds:add–spring dsmodel1 1
    (par–pos 0.5 0.5) (position 18 18 15) 100))
;; c1
; Get the tag type of the spring load.
(ds:get–tag–type dsmodel1 c1)
;; 4
; get the tags
(ds:get–tags dsmodel1)
;; (2 6 5 4 3 7)
```

ds:get–tan–display–gain

Scheme Extension: Deformable Surfaces

Action: Gets the display scaling value used to vary the visual length of all the tangent constraints' tangent vectors within the deformable model.

Filename: adm/ds_scm/dsscm.cxx

APIs: api_dm_get_attrib_dm2acis

Syntax: (**ds:get–tan–display–gain** owner [target=1])

Arg Types:

owner	entity
target	integer

Returns: real

Errors: None

Description: Gets the display scaling value used to vary the displayed length of all the tangent constraints' tangent vectors within the target deformable models. The default value is 1.0.

The target argument specifies which deformable model to use in a patch hierarchy. Valid values for target are:

1 = active deformable model
2 = root deformable model
–1 = active deformable model and offspring
–2 = root deformable model and offspring

Otherwise, the target is the deformable model whose tag identifier equals target.

owner ACIS face or edge on which the deformable model lives.

target specifies which deformable model to use in a patch hierarchy.

Limitations: None

Example:

```
; ds:get–tan–display–gain
; Build a test edge with some tag objects
; (6 control points, x side length = 36)
(define dsmodel1 (ds:test–edge 6 36 0))
;; dsmodel1
(define erase (entity:erase dsmodel1))
;; erase
; Don't render the edge.
; Add a position–tangent point constraint.
(define c1 (ds:add–pt–cstrn dsmodel1 2
    "pos_tan" (par–pos 0.5 0.5)) )
;; c1
; Render the loads, constraints, and curve
; normal–comb.
(ds:get–draw–state dsmodel1 1 )
;; 14
; Vary the tan_display_gain to change the image
; length of the constraint vector.
(ds:set–tan–display–gain dsmodel1 1 15.0)
;; ()
; Query the tan_display_gain.
(ds:get–tan–display–gain dsmodel1 1)
;; 15
```

ds:get–tight–state

Scheme Extension: Deformable Surfaces

Action: Gets the tight state of a load/constraint on a deformable model.

Filename: adm/ds_scm/dsscm.cxx

APIs: api_dm_get_attrib_dm2acis

Syntax: `(ds:get-tight-state owner tag)`

Arg Types:

owner	entity
tag	integer

Returns: integer

Errors: None

Description: When the tight state is 1, the load/constraint is tightened, overriding a default shape of 1. There is no effect when the default shape is 0.

The input argument owner is the face or edge being sculpted. tag is the identifier for the load/constraint to modify. When the tag identifier identifies a load/constraint, the deformable model in the patch hierarchy, which contains the load/constraint, becomes the active deformable model.

This extension returns a 1 or 0, corresponding to the tight state of the tag's tight state.

owner ACIS face or edge on which the deformable model lives.

tag identifier recognizes a constraint.

Limitations: None

Example:

```
; ds:get–tight–state
; Build a test square spline face.
; (6x6 control points, x and y side length = 36)
(define dsmodel1 (ds:test–face 6 6 36 36 0))
;; dsmodel1
; Clear the face.
(define erase (entity:erase dsmodel1))
;; erase
(define refresh (view:refresh))
;; refresh
; Render the loads and constraints.
(ds:set–draw–state dsmodel1 1
    (+ ds–draw–cstrns ds–draw–loads))
;; ()
; Make a curve constraint.
(define crv–cstrn (ds:add–str–cstrn dsmodel1 2 'p
     (par–pos .2 .2) (par–pos .8 .8)))
;; crv–cstrn
; Turn off tightening.
(ds:set–tight–state dsmodel1 crv–cstrn 0)
;; #t
; Check the tight state – it should be 0.
(ds:get–tight–state dsmodel1 crv–cstrn)
;; 0
; Some operations automatically reset the tight state
;   to 1 – curve tracking, for example.
(ds:make–tracking–curve dsmodel1 crv–cstrn)
;; 8
; We now have a tight state of 1.
(ds:get–tight–state dsmodel1 crv–cstrn)
;; 1
```

ds:get–type–string

Scheme Extension: Deformable Surfaces

Action: Returns a string describing the input type number.

Filename: adm/ds_scm/dsscm.cxx

APIs: None

Syntax: (**ds:get–type–string** type)

Arg Types:	type	integer

Returns: string

Errors: None

Description: Returns the string description for the input type value. The list of known type values includes:

–1	=	No such tag object
1	=	spline control point
2	=	point pressure load
3	=	distributed pressure load
4	=	spring load
5	=	spring set load
6	=	load curve
7	=	dynamic load
8	=	point constraint (not a seam)
9	=	seam curve constraint
10	=	curve constraint (not a seam)
11	=	seam curve constraint
12	=	deformable surface
13	=	deformable curve
14	=	vector load
15	=	attractor load
16	=	multi–surface link constraint
17	=	area constraint
18	=	link load
19	=	tracking position deformable curve
20	=	tracking tangent deformable curve
21	=	tracking curvature deformable curve

type is an input type value.

Limitations: None

Example:
```
; ds:get–type–string
; Get the string description for type value 7.
(ds:get–type–string 7)
;; "The Dynamic Load"
```

ds:get–xyz

Scheme Extension: Deformable Surfaces

Action: Computes an *xyz* position on a deformable model for an input par–pos location.

Filename: adm/ds_scm/dsscm.cxx

APIs: api_dm_get_attrib_dm2acis

Syntax: `(ds:get-xyz owner target=1 uv-position)`

Arg Types:

owner	entity
target	integer
uv–position	par–pos

Returns: position

Errors: None

Description: Computes an *xyz* position on a deformable model for a given uv–position input par–pos location. owner specifies the entity that owns the target deformable model.

The target argument specifies which deformable model to use in a patch hierarchy. Valid values for target are:

1 = active deformable model
2 = root deformable model
–1 = active deformable model and offspring
–2 = root deformable model and offspring

Otherwise, the target is the deformable model whose tag identifier equals target.

owner ACIS face or edge on which the deformable model lives.

target specifies which deformable model to use in a patch hierarchy.

uv–position is the u and v position values.

Limitations: None

Example:

```
; ds:get–xyz
; Add to a square test face a parabolic crv–load
; and use it.
; Build a test square spline face.
; (6x6 control points, x and y side length = 36)
(define dsmodel1 (ds:test–face 6 6 36 36 0))
;; dsmodel1
; Don't display entity / ds test face exists.
(define erase (entity:erase dsmodel1))
;; erase
; Render the loads and constraints.
(ds:set–draw–state dsmodel1 1
    (+ ds–draw–cstrns ds–draw–loads))
;; ()
; Evaluate a point on the face.
(ds:get–xyz dsmodel1 1 (par–pos 0.5 0.5))
;; #[position 18 18 1]
```