

# 3-D Data file formats and SPRING

January-2006

SPRING displays 3-D objects, allows manipulation of them, and simulates the “physics” of interactions between them. The 3-D objects include tissue models, surgical tools, and geometric elements such as spheres and planes.

Such objects have many attributes, the most basic of which is the 3-D geometry, which is defined by 3-D points connected by edges. These are then organized into triangular planar faces or tetrahedral elements. This essential geometric description may be created by specifying 3-D points and their connections by human specification, interactive 3-D drawing programs, from geometric algorithms, or by capturing the 3-D geometry from digitizer input or analysis of medical images from CT, MRI, or other modalities.

A connected set of points may describe a single simple object. A more entity comprised of articulated components requires more description, including the geometry of each subcomponent and the joints or hinges connecting them.

The geometric aspects of objects are often stored in formatted files in one of several types including SMF, WRL, OBJ, MESH, and other types.

Since 3-D objects can be described in a number of ways, SPRING accepts a variety of data commonly used data formats for such objects, allowing tissues, surgical tools, and other 3-D entities to be imported from other software and simulators. Such objects can become part of a reusable library, supporting the creation of new simulations.

This is a short description of the file formats. For all the details on parsing input files, consider reading the code for parsing input files. This is found in the SPRING application’s object.h/.cpp files. The routines that read the files are `ReadOBJ`, `ReadSMF`, `ReadVRML`, etc.

Data file formats supported by SPRING include:

File type	Extensions	Contains	Notes
<b>Amira</b>	.inp	Node data (x,y,z) Tetrahedron data Texture	Normal data in file is ignored, and normals are computed by SPRING.
<b>Mesh</b>	.mesh	Dynamics mode Material properties Node data (x,y,z), normals, ####, sub-object ID Texture Dynamics parameters	Sub-object ID labels each node, allowing hinged objects

<b>SMF</b>	. smf	Material properties Node data (x,y,z) Face information Texture	Derives edge data from the faces
<b>OBJ</b>	.obj	Material properties Node data (x,y,z) Faces Texture	Alias/Wavefront object format
<b>SOBJ</b>	.sobj	Description of a sphere or a rectangle in 3-D	SPRING object format - may be extended
<b>FRD</b>	.frd		
<b>VRML</b>	.wrl, .iv		
<b>CyberWare</b>	.cy		

### Data recognized by SPRING input:

SPRING recognizes some, but not necessarily all, of the possible data items in these formats. Below are descriptions of what SPRING will parse, and limitations or specializations on what SPRING will use.

#### AMIRA:

“AVS UCD (*Unstructured Cell Data*) format can be used to represent finite-element grids and associated data fields in 2 and 3 dimensions.”

See [http://amira.zib.de/mol/usersguide/HxFileFormat\\_AVSUCD.html](http://amira.zib.de/mol/usersguide/HxFileFormat_AVSUCD.html)

<http://www.amiravis.com/usersguide31/usersguide/index.html>

SPRING expects an ASCII file format:

N: # of nodes

N lines of node\_num, x, y, z, normal\_x, normal\_y, normal\_z

#### SMF:

Simple Model Format:

<http://www.csit.fsu.edu/~burkardt/data/smf/smf.txt>

Faces are always triangular.

# comments

v <x> <y> <z>

f <v1> <v2> <v3>

Restriction: SPRING does not use any of the attribute information on colors, normals, or textures. SPRING also ignores all grouping and transformation operators in this format.

**OBJ:**

Wavefront format. See [http://javaview.de//guide/formats/Format\\_Obj.html](http://javaview.de//guide/formats/Format_Obj.html)

```
# comments
v x y z      # 3-d coordinate of a vertex
vn x y z     # 3-d normal
vt x y       # 2-d texture coordinate
f int int int # refers to 3 or more vertices by position in file (starts with 1)
```

The *f* line values can include *v*, *vt*, and *vn* coordinates in several forms, including *v*, *v//*, *v//vn*, or *v/vt/vn*.

Basic SPRING does not handle *vn* and *vt* data correctly, and will fail if data of these types is present. SPRING only handles triangles, but will read 4 points, splitting into 2 triangles (1, 2, 3) and (1, 3, 4).

**SOBJ:**

```
// Reads an SOBJ file (=Spring OBJ) - The file contains commands
// to create the basic objects which are already defined in
// spring such as sphere and others.
```

```
sphere")) {
    // parse attributes
    float layers, radius, inner;
    sscanf(buffer, "%*s %f %f %f", &layers, &radius, &inner);

plane:
    sscanf(buffer, "%*s %f %f %d %d %f %f %f %c", &x_length, &y_length,
&x_steps, &y_steps,
&x_offset, &y_offset, &z_offset,
&orientation);
```

**FRD:**

Nodes are same as SMF

```
# comments
v <x> <y> <z>
e <n1> <n2>      # edges are defined here
f <1> <2> <3>    # faces are then read in
o <extrusion code> # extrusion information
d <dynamics code>
r <r>           # for diffuse color
g <g>
b <b>
a <alpha>
```

n <num faces>  
m <mass interval>  
c <spring constant>  
p <damping constant>

### **VRML**

Version 1.0 (<http://astronomy.swin.edu.au/~pbourke/dataformats/vrml1/>) and 2.0 (<http://www.graphcomp.com/info/specs/sgi/vrml/spec/>) files are read.

This is the most complex format that is handled. SPRING will handle multiple objects described here, but not all forms.