rsdlink

rsdlink converts 3D modeling data files (RSD files) to 3D modeling data files (TMD files) that can be handled by libgs.

The information below is for Version 3.72, 10/1/1996.

Command usage

Format

```
rsdlink [options] <RSD-names>...
rsdlink [options] <RSD-names> [options] <RSD-
 names> [options] ...
rsdlink [options] <arg-files>...
rsdlink [options] <prj-file>
```

Multiple RSD data specified as arguments is linked to create one TMD file. The file extension .RSD may be omitted. The scaling factor and parallel shift range can be set for each piece of RSD data (see Example 1).

When the argument RSD name does not include the path, first search the current directory, then search the ..\RSD directory.

When the argument is expanded, you may use the argument file (.ARG). Although the file extension (.RSD) can be omitted from the RSD file name, the argument file extension (.ARG) may not be omitted. (See Example 2.)

A project file (*.PRJ) may be allocated directly in the same fashion. In this case, the RSD described by the project file is the same as allocating the RSD entirely by arguments. It is treated at the same time as if the -id option were specified.

The project extension .PRJ may not be omitted.

Options

`-o` file-name	Specifies the output file name. The default is a.tmd.
`-s` factor	Scales the RSD data after the next argument (see note below).
`-sc` factor	The scaling factor is rounded up to 2n. The default is 1.0. In TMD format, the coordinates are 16-bit integers, so the appropriate scaling factor must be set within this coordinate system.
`-t xyz`	Performs parallel shift of the coordinate values after the next argument (see note below). The default is (0 0 0).
`-id` PRJ-file	An ID number array is formed for each object created in a TMD file by specifying the project file (model.prj) that controls the modelling data. The outcome is found in the current directory as a C language header file. The output file name is the TMD file name followed by .h as the extension (see Example 3).
`-info`	Outputs very detailed information, such as the type of objects being converted and vertex coordinates of the object being transformed, as well as texture information, to the standard output (see Example 5).
`-v`	Outputs detailed information about the conversion, such as the numbers of polygons, to the standard output. Approximate sizes (Range: minimum and maximum value of the vertices (x, y, z)) within the PlayStation coordinate system are also output to each RSD. The positions and sizes can be confirmed before displaying in the target box (see Example 4)

NOTE: -s and -t are valid for all RSD that subsequently appear until they are re-specified.

Main improvements from previous version (ver. 3.7)

- The bug which was causing incorrect output of the primary color vectors of the fourth vertex of quadrangles with light source calculation, no texture, being flat and with gradation has been rectified.
- The bug which was causing incorrect output of the normals of a triangle with light source calculation, no texture, being flat and with gradation has been rectified.

Restrictions

This version has the following restrictions.

- When there is an especially large number of polygons of one RSD data, sometime they cannot be linked. The upper limit of the number of polygons changes according to the number of vertices and the number of normal lines, etc., but assume it is approximately 5,000 polygons (only DOS version). However this limitation is with respect to one RSD, and there is no limit to the number of RSDs that may be linked, nor to the overall number of TMD poygons to be created.
- If you link an RSD which does not appear in the project file, -1 is assigned to the header file.
- If, when the project file is specified, RSD is specified more than twice, the ID value is assigned only to the first RSD and -1 is assigned to the second and subsequent RSDs.
- The first line of the input RSD file must be less than 1024 bytes.

Note

- When creating a program that uses the TOD file output by the layout tool and mktod, do not use options such as translation (-t) and scaling (-s). The extent of the move and the scaling factor of the RSD object are stored in the TOD file, so scaling cannot be performed smoothly when converting from RSD to TMD.
- Translation and scaling should be performed in RSD format using dxf2rsd (-sc -t) and rsdform (-s -t) etc. The data will be more accurate and easier to manage.
- Rsdlink performs file input/output in small units. As a result it can sometimes be extremely time consuming when specifying a file on the network as an input/output file. When linking large RSD data please use the local disk as much as possible.

Supplement

- The texture file (TIM file) described in the .RSD file must be in the same directory (.\) as the RSD file or in the ..\TIM directory. Rsdlink will search the TIM file in this order.

- The table below shows what TMD types rsdlink converts RSD into. The vertical and horizontal items are

  the types described in the MAT file of RSD (however, Tri, Quad, Line, Sprite are specified in the PLY file by numbers). 0x .... is the TMD packet header. The table also includes single-faced polygons that are not semi-transparent. Although the header value is different for semi-transparent and double-faced polygons, the packet is the same. See RSD.TXT and TMD.TXT for details.

Table 18-9:

RSD->TMD•Exchange Chart

	With Light Sourcing		Without Light Sourcing	
Shading	Flat	Gouraud	Flat	Gouraud
C+Tri	0x20000304	0x30000406	0x21010304	(0x21010304)
G+Tri	0x20040506	0x30040606	(0x31010506)	0x31010506
T+Tri	0x24000507	0x34000609	(0x25010607)	(0x25010607)
D+Tri	(0x24000507)	(0x34000609)	0x25010607	(0x25010607)
H+Tri	(0x24000507)	(0x34000609)	(0x35010809)	0x35010809
C+Quad	0x28000405	0x38000508	0x29010305	(0x29010305)
G+Quad	0x28040708	0x38040808	(0x39010608)	0x39010608
T+Quad	0x2c000709	0x3c00080c	(0x2d010709)	(0x2d010709)
D+Quad	(0x2c000709)	(0x3c00080c)	0x2d010709	(0x2d010709)
H+Quad	(0x2c000709)	(0x3c00080c)	(0x3d010a0c)	0x3d010a0c
C+Line	(0x40010203)	(0x40010203)	0x40010203	(0x40010203)
G+Line	(0x50010304)	(0x50010304)	(0x50010304)	0x50010304
T+Sprite	(0x64010305)	(0x64010305)	0x64010305	(0x64010305)
1x1	(0x6c010204)	(0x6c010204)	0x6c010204	(0x6c010204)
8x8	(0x74010204)	(0x74010204)	0x74010204	(0x74010204)
16x16	(0x7c010204)	(0x7c010204)	0x7c010204	(0x7c010204)

(First column represents material and polygon type)

Parentheses () enclose fundamentally unsuitable combinations of material types.

These are TMD that are considered to be suitable, ignoring partial specifications.

Usage Example 1 (Basic)

```
C:\> rsdlink -v -o boxes.tmd box1 -s 2.0 -t 100 100 100 box1
box2 -t 200 -200 200 box3
```

In this case, the following four objects become one TMD file (boxes.tmd).

box1 unity scaling factor

box1' doubles box1 (100 100 100) and shifts

box2 doubles box2 (100 100 100) and shifts

box3 doubles box3 (200 -200 200) and shifts

Usage Example 2 (Collecting Arguments in a File)

As in Example 1, this may be specified in the case of the expansion of an argument, when a file is gathered as below.

```
C:\> type test.arg (c  onfirms contents of .ARG file)
       box1
       -s 2.0 -t 100 100 100 box1 box2
       -t 200 -200 200 box3
C:\>rsdlink -v -o boxes.tmd test.arg (specifies .ARG
file)
```

Usage Example 3: (Creating an ID Array that Includes the Project File)

The project file (model.prj) is the ID table of objects created with the layout tool.

Example 3-1

```
C:\> type model.prj (checks contest of .PRJ file)
       @PRJ940701
       # Next available ID
       6
       # ID   Object Name
       0      head.rsd
       1      body.rsd
       2      l_arm.rsd
       3      r_arm.rsd
       4      l_leg.rsd
       5      r_leg.rsd
```

When there are files as above, and when you execute

```
C:\>rsdlink -o robo.tmd model.prj
```
(argument specification)

or

```
C:\>rsdlink -o robo.tmd -id model.prj body head l_arm r_arm l_leg
r_leg
```
(specifies -id option)

robo.tmd is created, and at the same time robo.h is created. (robo.h is explained below.) Each object's ID is recorded as a component of the array in the order that it appears in TMD. You can use this file when you access each object within TMD from the application.

Example 3-2: robo.h Listing

```
/*
*      TMD ID Lis t for robo.tmd
*
*      Created by rsdlink (v3.1) at 10:10 06/16/94
*
*      (C) 1994 Sony Computer Entertainment Inc. All Rights
Reserved.
*/
       int robo_list[] = {
              1,     /* body */
              0,     /* head */
              2,     /* l_arm */
              3,     /* r_arm */
              4,     /* l_leg */
              5,     /* r_leg */
};
```

Example 4: (Output Example When Using -v Option)

The -v option provides the following output.

Example 4

```
C:\> rsdlink -v dino -s 100 box
```

Output	Explanation
`[0]================ RSD ================`	first RSD (DINO.rsd)
`  RSD files : \PSXGRAPH\DATA\RSD\dino.ply,` `             DINO.mat, DINO.grp`	
`       TEX[0] = dino0. tim`	
`       TEX[0] = dino1. tim`	
`       TEX[2] = dino2. tim`	texture file name
`       TEX[3] = dino3. tim`	
`       TEX[4] = dino4. tim`	

TEX[5] = dino5. tim	
POLYGON : 2724	number of polygons
VERTEX : 1376	number of vertices
NORMAL : 2671	number of normal lines
MATERIAL : 2592	amount of material
Range : (-180, -210, -1690)-(180, 580, 290)	minimum and maximum of (x,y,z)
[1]================ RSD ================	
RSD files : \PSXGTSPH\DATA\RSD\BOX.ply, BOX.mat,BOX.grp	second RSD (BOX.rsd)
POLYGON : 12	
BERTEX : 8	
NORMAL : 12	
MATERIAL : 1	
Scale : 128	unity scaling factor (100 is rounded up to 2n)
Range : (-6400, -6400, -6400)-(6400, 6400, 6400)	
================ TMD ================	
Output TMD : “a.tmd”	output file name
TMD header : (12 bytes)	TMD file header size
Objects : 2(56 bytes)	total number of RSD
Primitives : 2736(65640 bytes)	total no. of primitives
12 ...Flat Colored Triangles	
136 ...Gouraud Colored Triangles	contents of each mode
2434 ...Flat Textured Triangles	
154 ...Gouraud Textured Triangles	
Vertices : 1384 (11072 bytes)	total number of vertices
Normals : 2683 (21464 bytes)	total no. of normal lines

Total file size: 98244 bytes	<output file size>

Example 5: (Example of Output Using the -info Option)

It is possible to confirm the contents of TMD data that has been changed by using the -info option.

Example 5

```
C:\> rsdlink -info box
```

Output	Explanation
Input RSDs : 1 object(s)	number of objects in TMD
RSD[0] = “BOX”	RSD name
Total vertices : 8	total number of vertices
Total normals : 12	total number of normal lines
Total primitives : 12	total number of primitives

```
-----------------<box>----------------
= [ 0] FLAT TEX 3-POLY (0x24000507) LIGHT: ON =
      Vert-0: (   -150,   -150,   -150)  (#2)
      Vert-1: (   -150,   -150,   -150)  (#6)
      Vert-2: (   -150,   -150,   -150)  (#0)
      Norm-0: (      0,      0,  -4096)  (#0)
      UV 0-2: ( 0 0) (47 0) ( 0 47)
      Pixel mode  : 4bit CLUT : (x,y)=( 0 480)
      Texture Page: 10  Texture No. : 0
= [1] FLAT TEX 3-POLY (0x24000507) LIGHT: ON=
                .
                .
                .
```

Primitive Information

Each primitive displays information in the following fashion.

: Primitive Information

```
=[ 0] FLAT TEX 3-POLY (0x24000507) LIGHT: ON = 1 2 3 4 5 6
```

1	:	Polygon number
2	:	FLAT/GOURAUD
3	:	TEX/NO TEX ON/OFF (TRANS) TWO-SIDED GRADATION
4	:	Primitive types (3-POLY, 4-POLY, LINE, SPRITE)
5	:	Display of 16 consecutive primitive headers (0x...)
6	:	light source calculations (LIGHT : ON/OFF).

Vertex Coordinate Values

The coordinates of each vertex, which consists of these primitives, are displayed as shown below. (#...) is the vertex number used in the PLY file.

```
Vert-0: (-150, -150 -150) (#2)
```

Normal Lines

Normal lines are displayed in a similar manner. In RSD, Normal lines are usually regulated in size 1, but (in this case (0, 0, -1.0)) in TMD, conversion calculations treat 4096 as 1.

```
Norm-0: ( 0, 0, -4096) (#0)
```

Other Information

Textures, UV coordinates, and material information such as color are then displayed.