

零基础学习ejoy2d——ppm贴图

gaccob

2004 年 2 月 5 日

1. ppm贴图格式简介

- pbm, portable bitmap, 单色图(1 bit).
- pgm, portable gray map, 灰度图.
- ppm, portable pixel map, 真彩图.

pbm / pgm / ppm 图像的文件格式分为两部分： 文件头和数据部分。
一个典型ppm头的sample:

```
1  p6 # ppm格式
2  1024 1024 # 高, 宽
3  255 # 深度, 不一定是255
```

ppm的格式有p3和p6, p3表示用ascii码(文本)来表示数据, p6表示以字节码(二进制)来表示, 每一个像素按(r, g, b)的格式来存储.

```
1  P3
2  4 4
3  15
4  0 0 0 0 0 0 0 0 0 15 0 15
5  0 0 0 0 15 7 0 0 0 0 0 0
6  0 0 0 0 0 0 0 15 7 0 0 0
7  15 0 15 0 0 0 0 0 0 0 0 0
```

pgm与ppm类似, 格式有p2和p5, p2表示文本, p5表示二进制.

```
1  P2
2  18 7
3  15
4  0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
5  0 3 3 3 3 0 0 7 7 7 7 0 0 11 11 11 0
```

```

6      0 3 0 0 0 0 0 7 0 0 0 0 0 11 0 0 0 0
7      0 3 3 3 0 0 0 7 7 7 0 0 0 11 11 11 0 0
8      0 3 0 0 0 0 0 7 0 0 0 0 0 11 0 0 0 0
9      0 3 0 0 0 0 0 7 7 7 7 0 0 11 11 11 11 0
10     0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0

```

对于pbm来说, 格式为p1, 但是文件头中没有最大颜色, 因为用0和1来表示就可以了.

```

1      P1
2      24 7
3      0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
4      0 1 1 1 1 0 0 1 1 1 1 0 0 1 1 1 1 0 0 1 1 1 1 0
5      0 1 0 0 0 0 0 1 0 0 0 0 0 1 0 0 0 0 0 1 0 0 1 0
6      0 1 1 1 0 0 0 1 1 1 0 0 0 1 1 1 0 0 0 1 1 1 1 0
7      0 1 0 0 0 0 0 1 0 0 0 0 0 1 0 0 0 0 0 1 0 0 0 0
8      0 1 0 0 0 0 0 1 1 1 1 0 0 1 1 1 1 0 0 1 0 0 0 0
9      0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0

```

参考文章: <http://www.cppblog.com/windcsn/archive/2005/11/11/ppm.html>

2. ejoy2d中的ppm源码阅读

ejoy2d中的ppm贴图的处理都在lib/ppm.h和lib/ppm.c中. ejoy2d中用pgm贴图来描述alpha通道, 用ppm贴图来描述rgb通道.

```

1      struct ppm {
2          // 指ppm贴图的格式, p1-p6分别对应1-6
3          int type;
4          // 图像的深度, 一般有255(8位)和15(4位)
5          int depth;
6          // 步长, alpha是1, rgb是3, rgba就是4
7          int step;
8          // 长 & 宽
9          int width;
10         int height;
11         // 图像数据
12         uint8_t *buffer;
13     };
14
15     // 载入ppm文件头
16     static int
17     ppm_header(FILE *f, struct ppm *ppm) {
18         .....
19     }
20

```

本着自由的精神, 本文档可以随意阅读, 修改, 发布; 如涉及相关引用的版权问题, 请联系gaccob@qq.com及时修改.

```
21 // 载入ppm文件数据
22 // 这里会根据type(p1, p2, ...)的不同, 做对应的解析并载入
23 // skip是为了有一个初始offset(适用于rgb ppm载入alpha的情况)
24 static int
25 ppm_data(struct ppm *ppm, FILE *f, int id, int skip) {
26     .....
27 }
28
29 // 载入ppm文件, 调用ppm_header()和ppm_data()完成.
30 // 如果是rgba, 需要从两个贴图文件一起载入(会做一致性校验)
31 static int
32 loadppm_from_file(FILE *rgb, FILE *alpha, struct ppm *ppm) {
33     .....
34 }
35
36 // 载入ppm文件的lua接口
37 // lua输入参数:
38 //   string ppm_name
39 // 输出lua结果:
40 //   string format(这里约定的格式有: RGBA8, RGB8, ALPHA8, RGBA4, RGB4,
41 //                 ALPHA4)
42 //   int width
43 //   int height
44 //   table buffer(ppm数据部分)
45 static int
46 loadppm(lua_State *L) {
47     .....
48 }
49
50 // 载入ppm文件到texture(纹理)的lua接口
51 // lua输入参数:
52 //   string ppm_name
53 // lua输出参数:
54 //   TODO: 这个到后面讲texture时再看
55 static int
56 loadtexture(lua_State *L) {
57     .....
58 }
59
60 // 根据format(上面描述的RGBA8等格式), 设置ppm数据: type, depth, step
61 static void
62 ppm_type(lua_State *L, const char * format, struct ppm *ppm)
63 {
64     .....
65 }
66
67 // 从lua中读取数据, 保存rgb的ppm贴图(写文件), P6二进制格式.
68 static void
69 save_rgb(lua_State *L, int step, int depth) {
```

```
68     .....
69 }
70
71 // 从lua中读取数据, 保存alpha的pgm贴图(写文件), P5二进制格式.
72 static void
73 save_alpha(lua_State *L, int step, int depth, int offset) {
74     .....
75 }
76
77 // 保存ppm文件的lua接口, 调用save_rgb()和save_alpha()实现, 写文件.
78 // lua输入参数:
79 //   string save_filename(保存的文件名)
80 //   string format(同上)
81 //   int width
82 //   int height
83 //   table buffer(ppm数据部分)
84 static int
85 saveppm(lua_State *L) {
86     .....
87 }
88
89 // lua的导出接口
90 int
91 enjoy2d_ppm(lua_State *L) {
92     luaL_Reg l[] = {
93         { "texture", loadtexture },
94         { "load", loadppm },
95         { "save", saveppm },
96         { NULL, NULL },
97     };
98     luaL_newlib(L, l);
99     return 1;
100 }
```