OpenCascade Chinese Text Rendering

eryar@163.com

Abstract. OpenCascade uses advanced text rendering powered by FTGL library. The FreeType provides vector text rendering, as a result the text can be rotated and zoomed without quality loss. FreeType also support unicode charset. The paper focus on the Chinese Text rendering.

Key Words. OpenCascade, FreeType, Chinese Text, 中文汉字, Unicode

1. Introduction

OpenGL 中并没有提供直接的文字绘制支持，一个通用的二维文字解决方案是使用 glDrawPixels()来显示位图形式的字体，前提是用户已经预先生成了一系列的位置形式的文字字库，这也是很多早期计算机游戏的通用做法。使用位图来绘制文字的主要问题是不能控制显示过程中的图像走样。因为文字图像的大小总是一定的，缩放后变形比较明显；如果图像是根据视点实时地进行缩放，则势必消耗大量的系统资源。一个较好的解决方案就是使用纹理来表达矢量类型的文字。矢量文字的优点在于：每个字型都是使用数学公式来描述的，并使用光滑的曲线实现笔画之间的连接。因此，将矢量文字用纹理来表达的话，只要预先设置的纹理分辨率满足需求，那么对纹理面进行缩放或改变用户的视点时，都不会造成明显的文字失真变形。

矢量文字的处理首推著名的开源跨平台开发库 FreeType。这是一个专业的字体数据解析工具，可以解析 TrueType, Type1 等多种矢量字体格式，并通过统一的函数接口提供给用户程序使用。FreeType 本身不包含文字排版和图形化显示的功能，因此可以直接将它解析字体文件的结果应用在 OpenGL 程序中。

OpenCascade 的文字显示就用到了 FreeType 库，将文字转换成了矢量图形，所以可以对其任意缩放，都不会影响其显示质量。且还支持 Unicode 的文字的显示，当然也包含中文的显示。本文主要介绍在 OpenCascade 中显示中文的注意事项，也介绍了 OpenCascade 中将文字转换成 TopoDS_Shape 的功能。

2. Render Chinese Text

OpenCascade 在 Draw Test Harness 中有关于显示文字的命令 vdrawtext，显示文字效果如下图所示：

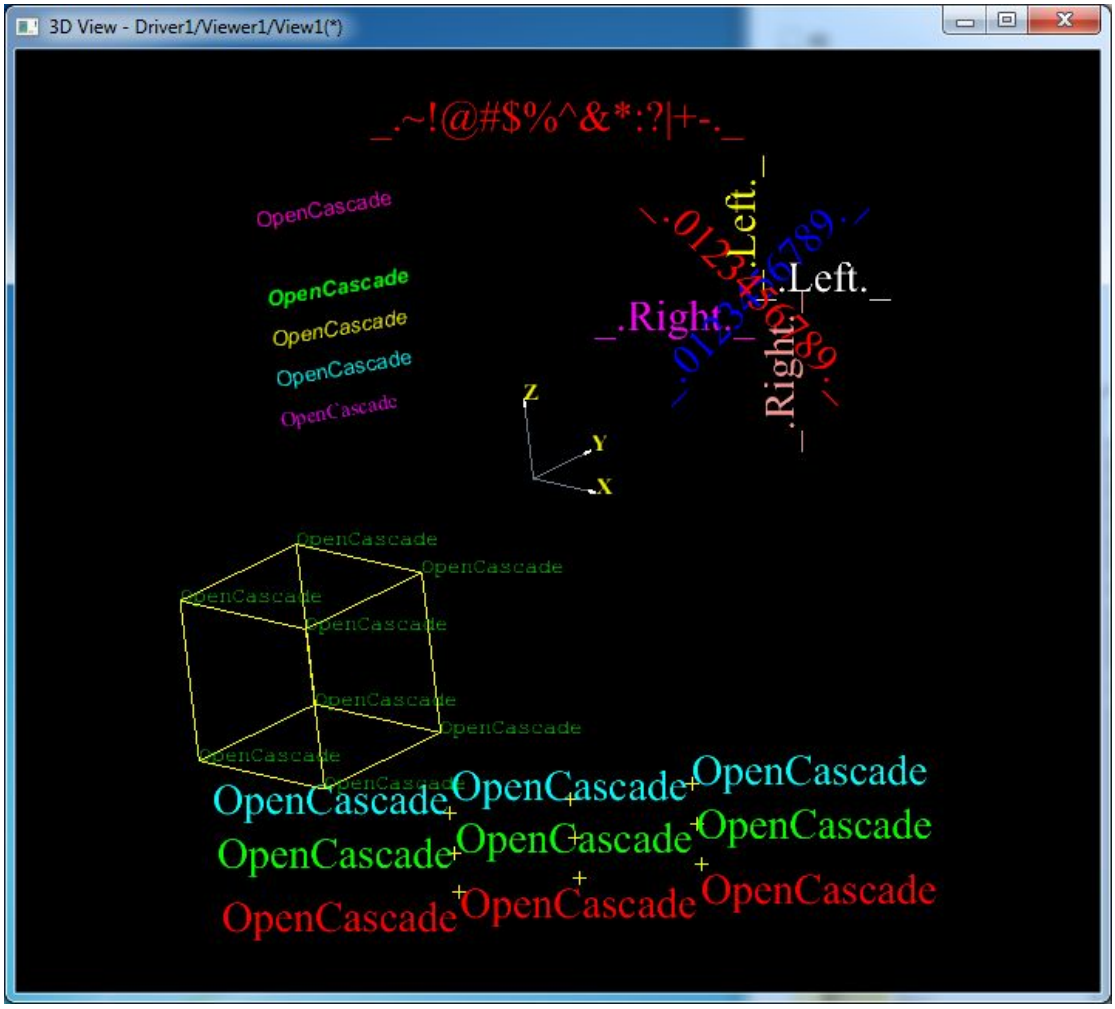

Figure 2.2 Text in Draw Test Harness

实现上图的 Tcl 命令如下图所示：

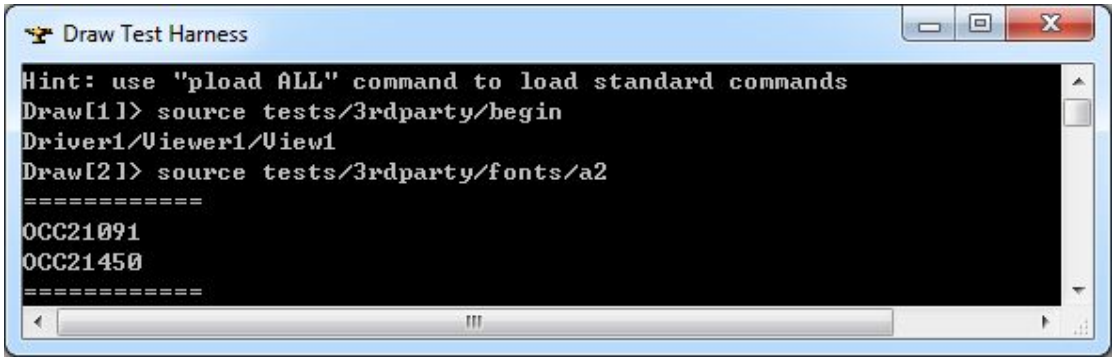

Figure 2.2 Draw Text Tcl Command

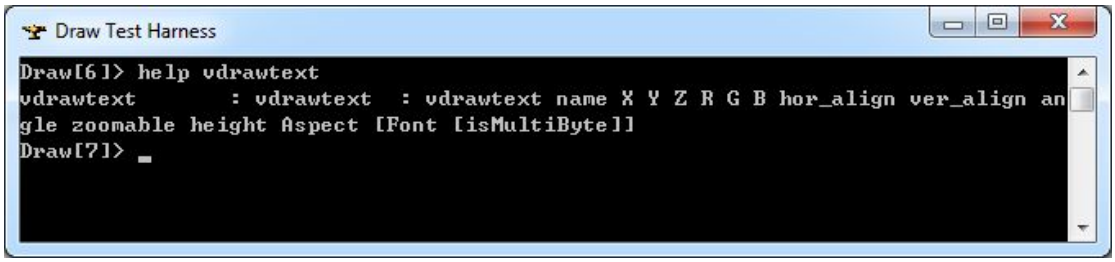

Figure 2.3 vdrawtext command

从 vdrawtext 命令中可以看出，最后一个参数就是关于多字节字符串的显示处理。输入如下命令来显示包含中文的字符串：

```
vdrawtext  你好 OpenCascade 100    300 -400 000 255 255 0 0 000 1 50 1 SimSun 1
```

显示结果如下所示：

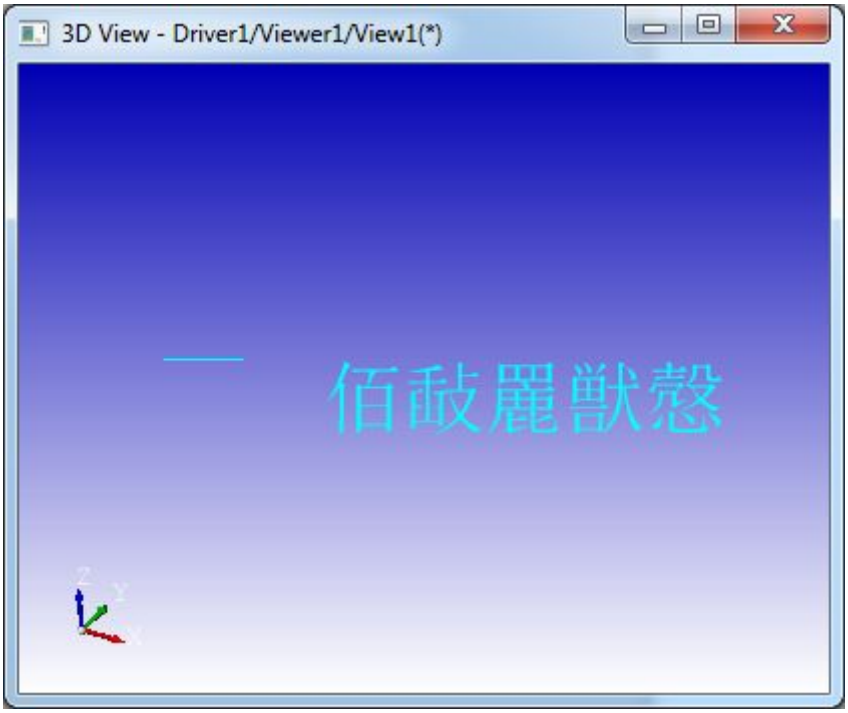

Figure 2.4 Render Chinese Text by vdrawtext command

由图可知，显示结果不正确。找到 vdrawtext 命令实现部分的源代码，实现代码在文件 VeiwerTest_ObjectCommands.cxx 中，修改其转换算法后代码如下所示：

```
static int VDrawText (Draw_Interpretor& di, Standard_Integer argc, const char** argv)
{
  // Check arguments
  if (argc < 14)
  {
    di<<"Error: "<<argv[0]<<" - invalid number of arguments\n";
    di<<"Usage: type help "<<argv[0]<<"\n";
    return 1; //TCL_ERROR
  }

  Handle(AIS_InteractiveContext) aContext = ViewerTest::GetAISContext();
```

```
// Create 3D view if it doesn't exist
if ( aContext.IsNull() )
{
   ViewerTest::ViewerInit();
   aContext = ViewerTest::GetAISContext();
   if( aContext.IsNull() )
   {
      di << "Error: Cannot create a 3D view\n";
      return 1; //TCL_ERROR
   }
}

// Text position
const Standard_Real X = Draw::Atof(argv[2]);
const Standard_Real Y = Draw::Atof(argv[3]);
const Standard_Real Z = Draw::Atof(argv[4]);
const gp_Pnt pnt(X,Y,Z);

// Text color
const Quantity_Parameter R = Draw::Atof(argv[5])/255.;
const Quantity_Parameter G = Draw::Atof(argv[6])/255.;
const Quantity_Parameter B = Draw::Atof(argv[7])/255.;
const Quantity_Color aColor( R, G, B, Quantity_TOC_RGB );

// Text alignment
const int hor_align = Draw::Atoi(argv[8]);
const int ver_align = Draw::Atoi(argv[9]);

// Text angle
const Standard_Real angle = Draw::Atof(argv[10]);

// Text zooming
const Standard_Boolean zoom = Draw::Atoi(argv[11]);

// Text height
const Standard_Real height = Draw::Atof(argv[12]);

// Text aspect
const Font_FontAspect aspect = Font_FontAspect(Draw::Atoi(argv[13]));

// Text font
TCollection_AsciiString font;
if(argc < 15)
   font.AssignCat("Courier");
```

```
  else
    font.AssignCat(argv[14]);

  // Text is multibyte
  const Standard_Boolean isMultibyte = (argc < 16)? Standard_False : (Draw::Atoi(argv[15]) !=
0);

  // Read text string
  TCollection_ExtendedString name;
  if (isMultibyte)
  {
      /* eryar modified 20140817 11:11
    const char *str = argv[1];
    while ( *str || *(str+1)=='\x0A' || *(str+1)=='\x0B' || *(str+1)=='\x0C' || *(str+1)=='\x0D'
                    || *(str+1)=='\x07' || *(str+1)=='\x08' || *(str+1)=='\x09' )
    {
      unsigned short c1 = *str++;
      unsigned short c2 = *str++;
      if (!c2) break;
      name += (Standard_ExtCharacter)((c1 << 8) | c2);
    }
    */
      Resource_Unicode::ConvertGBToUnicode(argv[1], name);
  }
  else
  {
    name += argv[1];
  }

  if (name.Length())
  {
    Handle(MyTextClass)          myT          =          new
MyTextClass(name,pnt,aColor,hor_align,ver_align,angle,zoom,height,aspect,font.ToCString());
    aContext->Display(myT,Standard_True);
  }

  return 0;
}
```

主要是当是多字节字符串，使用 Resource_Unicode::ConvertGBToUnicode()函数来实现字符串转换，修改后仍输入前面的命令，可以显示中文字体了：

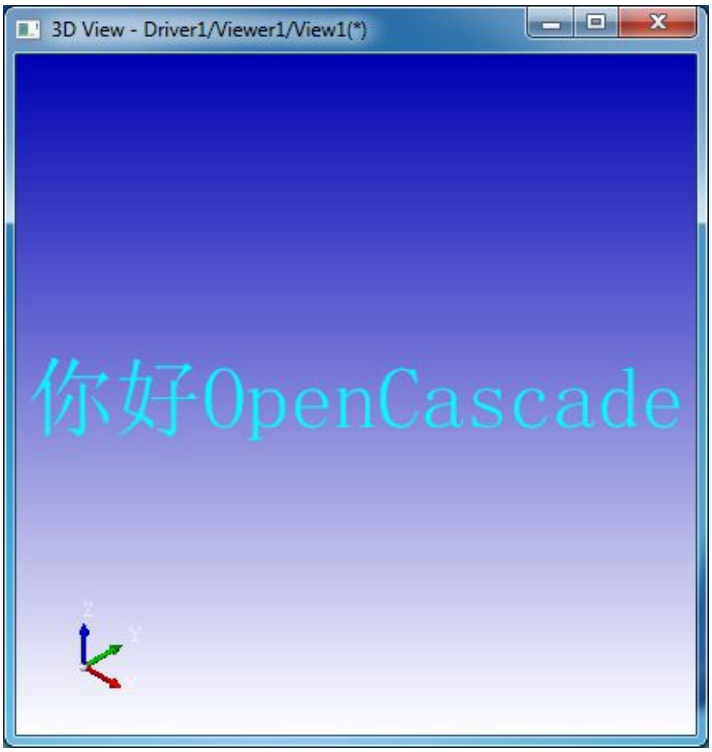

Figure 2.5 Render Chinese Text by vdrawtext command

综上所述，结合 Draw 中的代码可知，要在 OpenCascade 中显示中文，需要注意以下几点：

- 由于 OpenCascade 并没有提供直接显示文字的类，都需要从 AIS_InteractiveObject 派生一个文字显示类，并重载有关函数 Compute()；
- 字符串转换要使用 Resource_Unicode::ConverteGBToUnicode()来将中文的字符串转换为 Unicode 字符串；
- 一定要选择正确的中文字体，否则也是显示不正确的。

Figure 2.6 A Chinese Quote

3. Convert Text to TopoDS_Shape

借助于 TreeType 库 OpenCascade 可以将文字转换成样条并生成 TopoDS_Shape，即三维文字效果，相关的 draw 命令是 text2brep，生成效果如下图所示：

Figure 3.1 3D Text in Draw Test Harness

生成上述效果的 Tcl 脚本如下所示：

```
text2brep text2d eryar@163.com Times-Roman 18 bold composite=0
prism text text2d 0 0 2
vdisplay text
vsetdispmode 1
vfit
```

4. Conclusion

OpenCascade 使用 FreeType 来实现了文字的高质量的显示，因为是矢量图形，所以任意缩放不影响文字的质量。

OpenCascade 中显示中文时，需要注意字符串转换到 Unicode 时选择正确的转换函数，且要选择正确的字体格式，即中文字体，本文中仅以仿宋 SimSun 为例。

OpenCascade 还可将文字转换为 TopoDS_Shape 进而可以显示三维字体。

综上所述，可知 FreeType 库的功能还是很强大的。

5. References

1. 王锐，钱学雷，OpenSceneGraph 三维渲染引擎设计与实践，清华大学出版社
2. 在 OpenCasCade 的 2D 窗口中显示汉字的方法,
   http://www.cadcaecam.com/forum.php?mod=viewthread&tid=15444
3. OpenCascade Draw Test Harness code