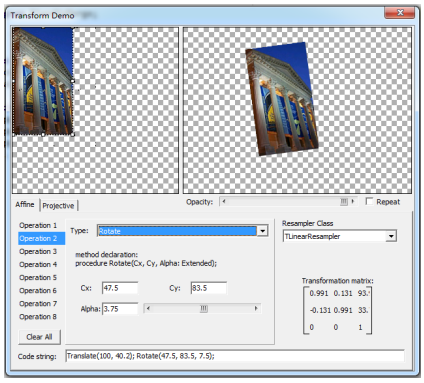
MemoryRaster 内存光栅引擎：投影

投影是高级光栅概念，在任何光栅系统中，凡是出现了投影(projection)功能的地方，都或多或少搭配了高级的框架。

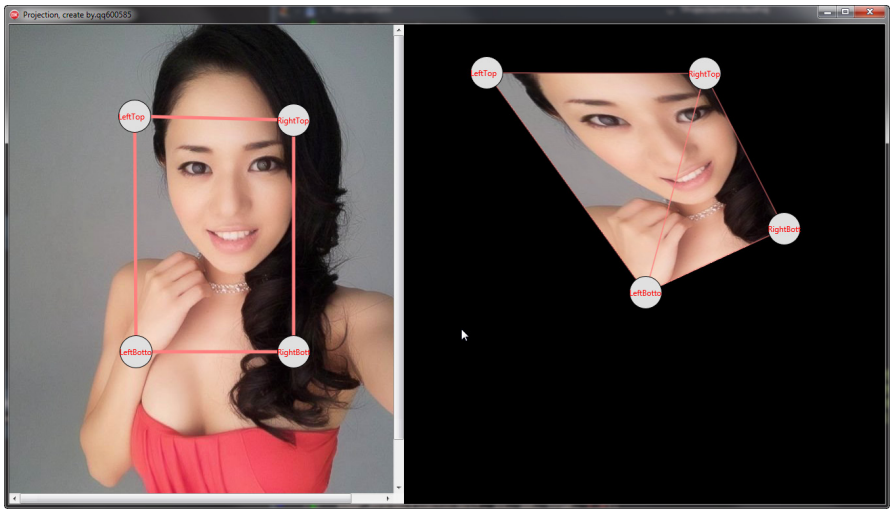
下面我们来对比一下传统的低级矩阵变换方式，这是 Graphics32 的 transform 的演示，通过旋转，平移，放大，三要素，构建了 3*3 对称坐标换算矩阵，通过对称矩阵可以从正反两个方向来求取坐标系，达到重构图像的目的。



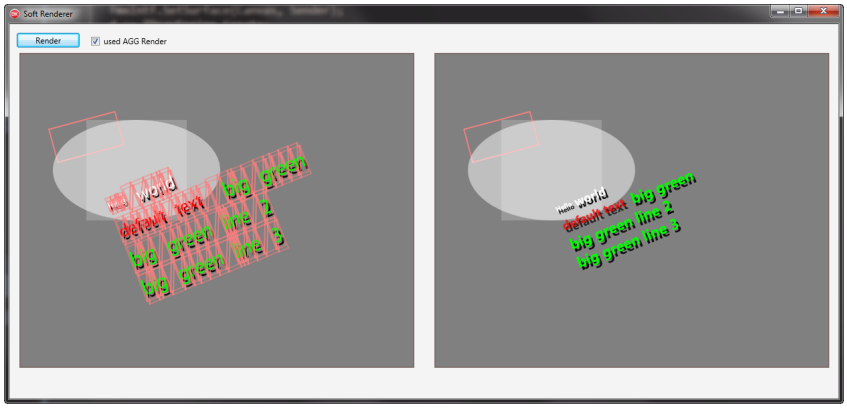
在 MemoryRaster 中没有对称矩阵这种概念，因为作者觉得的使用矩阵转换坐标系太绕，会让很多高级坐标系的代码看起来太蠢，假如由 4 顶点结构组成的体系 TV2Rect4 使用矩阵来转换坐标系，就会失去它的作用：本身设计 TV2Rect4 坐标系就是用于三角拆分的，在 opengl,vulkan,等等专业绘图体系，原子几何都是三角型，很老的绘图体系，类似 d2d，是使用矩阵做框坐标系变换，在 dx9 以后而 d2d 所依赖绘图 api，其实都是 d3d，换句话说，d2d 使用矩阵做框转换，然后再转换成 d3d 的三角，这时候图片才被渲染渲染出来。在手机上，亦然如此，opengles，都是原子三角型做图片填充，叫做顶点。

从下图中可以看出明显差异，MemoryRaster 没有使用矩阵，所有矩阵支持旋转，放大，平移，变形，等等变换在这里均使用 TV2Rect4 坐标系来表达，最后，应用性能，表现能力，等等效果远远超过矩阵坐标系变换，符合正常人类的直观理解。

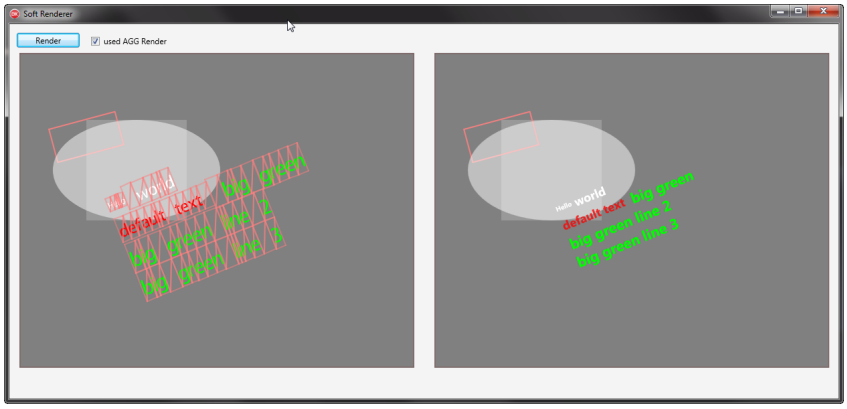
我们通过 Geometry2Dunit 可以查看到 TV2Rect4 的主结构，它非常简单，所有的变换就是修改 4 个顶点的坐标。最关键的是，使用 TV2Rect4 坐标来描述的图形，有 direct map GPU 的能力，这些都是图形领域的地基，不能有水分。



在 MemoryRaster 中，DrawText 都是投影的图形，换句话说，我们在每画一个字，都会有两个三角形通过投影才被画出来。
下图，左边使用了 MemoryRaster 作为软渲染输出
因为打开的字体影子，所有的文字被画了 2 次，所以我们才看见有很多三角形叠加。其实每一个字，只有两个三角形。



关闭影子捕获后，这样看更加直观



这里我们从投影角度，来理解两个 FMX 接口库对 zDrawEngine 的支持效果

zDrawEngineInterface_FMX：这个库的重点是在手机上高速绘图支持，使用它需要修改一些 FMX 的内置代码，像打补丁一样。在手机上，都是 opengles 的绘图接口，因为除了 3d 透视和正交，其它地方如果使用矩阵变换真的太费了。该接口会直接绕过矩阵变换，直接使用两个三角形来画出一张图片，它的概念，与 MemoryRaster+TV2Rect4 投影是一致的，因为三角是专业图形 api 的原子结构。而在 windows 平台上并不会使用三角，因为 FMX 在 windows 默认使用 d2d 画图（反锯齿效果好），没必要使用三角结构。

zDrawEngineInterface_SlowFMX：这个库没有使用三角画图片的概念，它构建了一套可以和 FMX 内置的 3x3 矩阵对齐计算的矩阵，每画一张图前都会重新计算一次矩阵变换。因此，在手机平台的性能下降 30%，pc 硬件因为 cpu 主频高，无感觉，即使画 10000 张图，也是 60 满帧率。

总结：在 MemoryRaster 中，三角形式的投影无处不在，无处不用。2D 坐标系支持都在 Geometry2Dunit 这个库里面。

2019-7
By.qq600585