GsTriSubdivide: Test Application Explanation;

INI file: GsTriSubdivide.ini

[INITVALS]

TOTALBUFFERS=1

TOTALSTREAMOUTBUFFERS=1

FXFILENAME="..\GsTriSubdivide\Source\GSTriSubdivide.fx"

DRAWNOSUBDIVIDETECHNIQUENAME="NoGSTriSubdivide"

DRAWTECHNIQUENAME="GSTriSubdivide"

STREAMOUTTECHNIQUENAME="StreamOut"

POINTSPERBUFFER=30000

MAXDRAWCALLS=100000

DRAWCALLSTEP=0

DRAWCALLSTART=1

//Line == 2, Loop == 1, Culdisack == 0

LOOPORCULDISACK=2

LOCKCAMERA=0

**Memory Size**

IN: Vertex: POS,COLOR == size of 28 == (7\*4 bytes):

STREAMOUT:Vertex: POS,COLOR== size of 28 == (7\*4 bytes):

DRAW:Vertex:POS,TEXCORD,COLOR == size of 44 == (11\*4 bytes):

STREAMOUT LOOP Memory size == STREAMOUT:Vertex X Number of Vertices being Drawn from the IN:VertexBuffer \* 3

DRAW OUTPUT Memory size =< DRAW:Vertex \* Number of Vertices being Drawn\* (3)(for LOOP and 1 if Culdisack)) \*3.

**A couple of examples to help:**

Ini settings:

Pointsperbuffer = 10000

DrawCallStart=10000

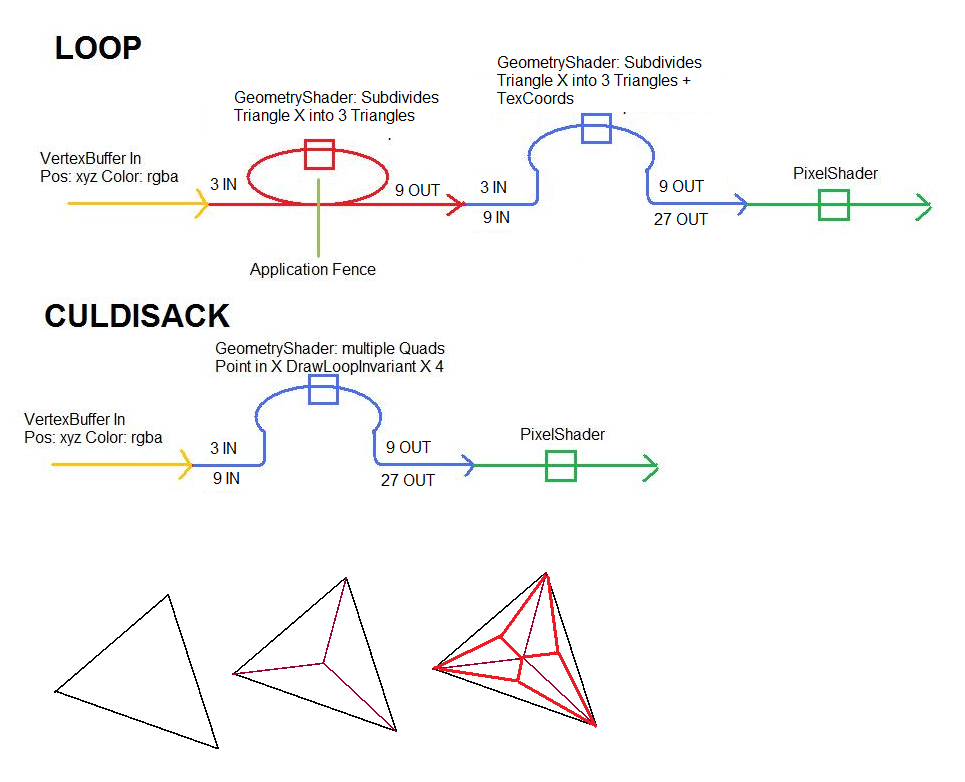
LoopOrCuldisack=0 //set to culdisack

Draw Memory Size would be: 44 X 10000 X 3 = 1320000 bytes or 1.32MB

If we change LoopOrCuldisack=1 //set to Loop

StreamOut Memeory size would be: 28 \* 10000 \* (3) = 840000 bytes or 0.84MB

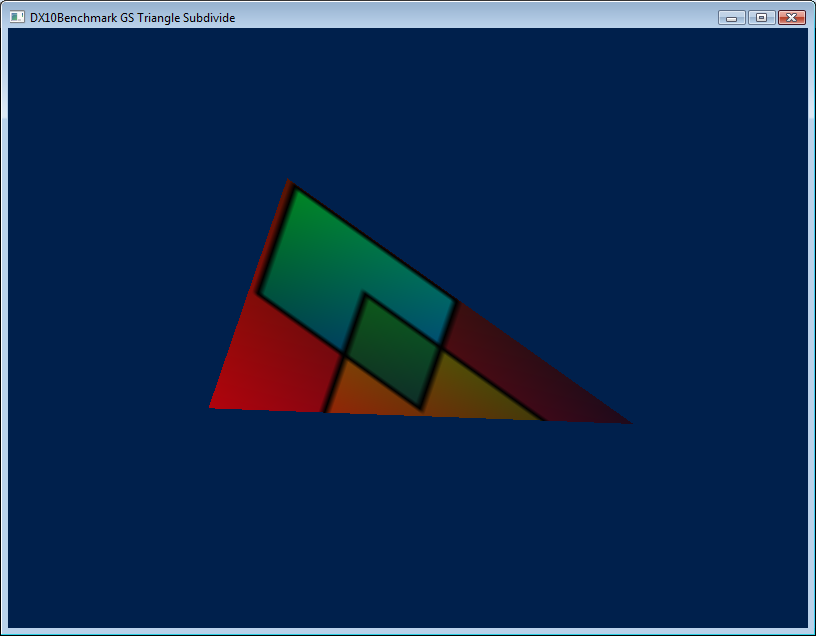
Draw Memory Size would be: 44 \* 10000 \* 3 \* 3 = 3960000 bytes or 3.96MB

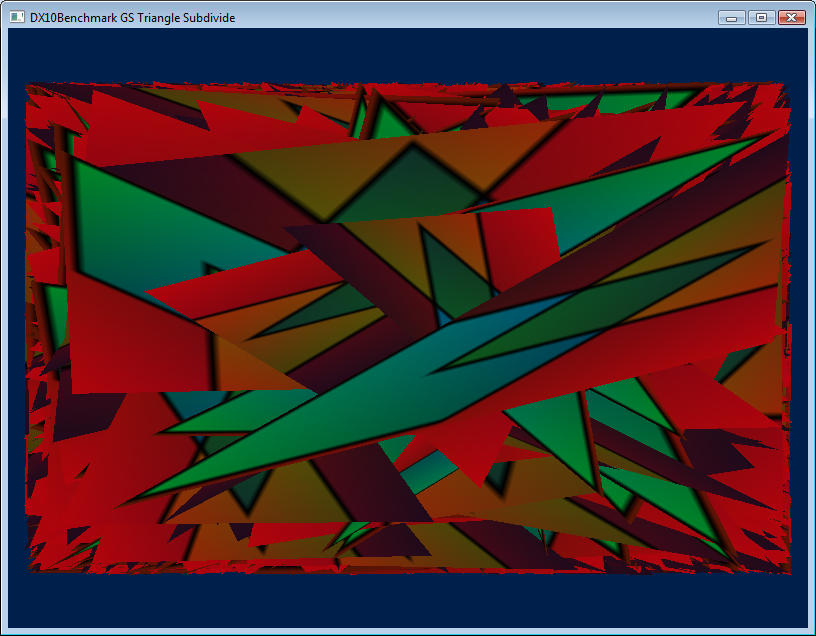
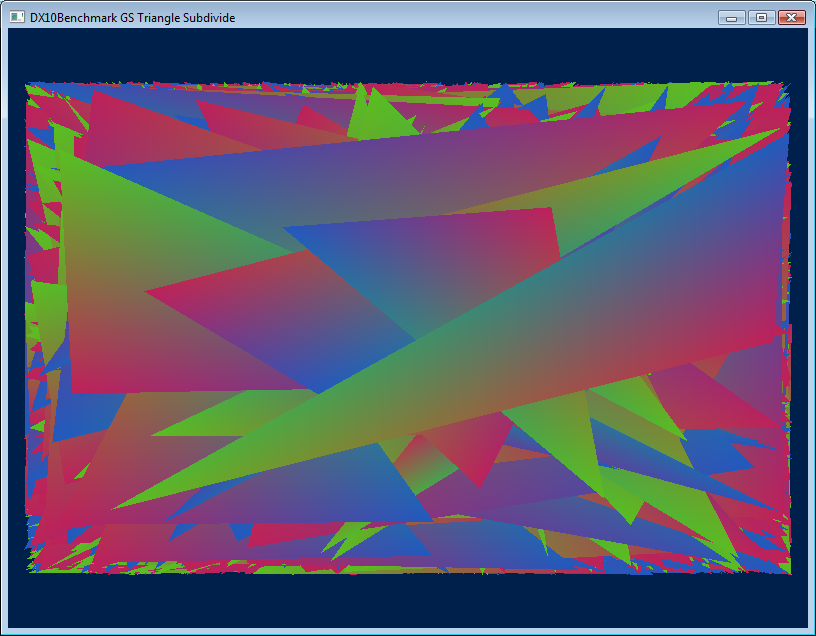


Triangle No Subdivide:

INI File

LOOPORCULDISACK=2



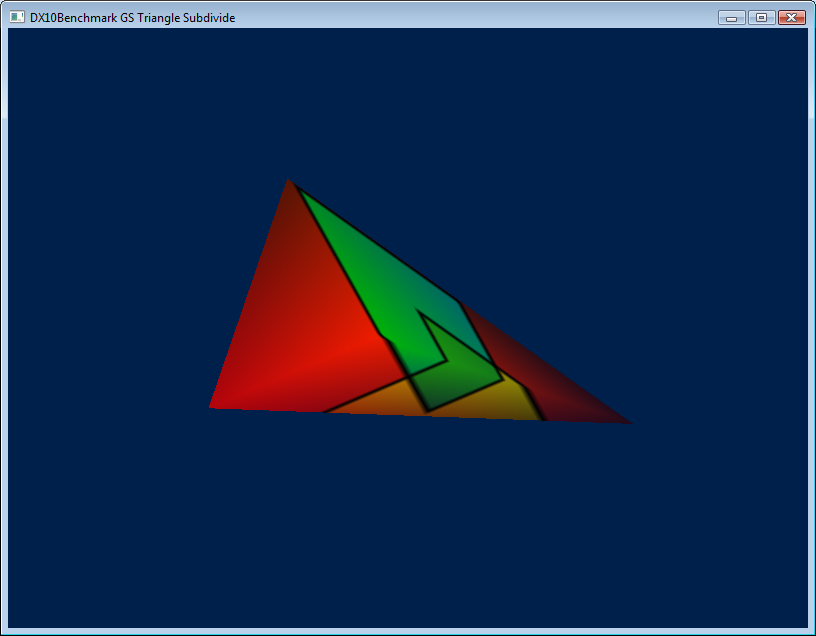
****

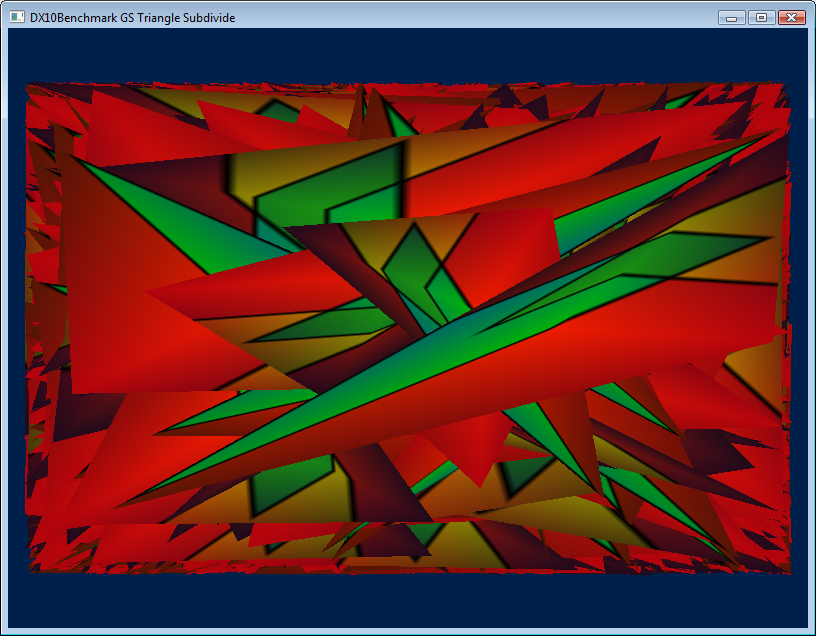
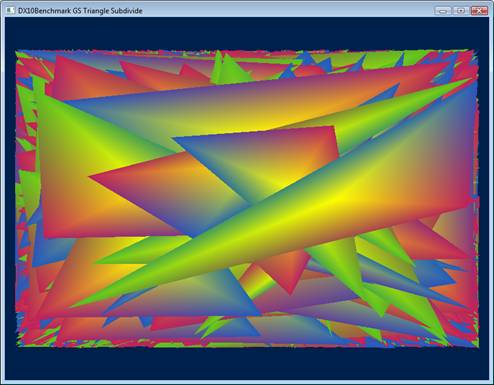
Culdisack with 3 Vertices in the Buffer: 3 Triangles out (9 vertices)

INI: LOOPORCULDISACK=0, POINTSPERBUFFER=3,DRAWCALLSTART=1,DRAWCALLSTEP=0,MAXDRAWCALLS=20000

LOOP with 3 Vertices in the Buffer: 3 Triangles out (9 vertices)

The first one you can see that we do a subdivision of a triangle into three triangles



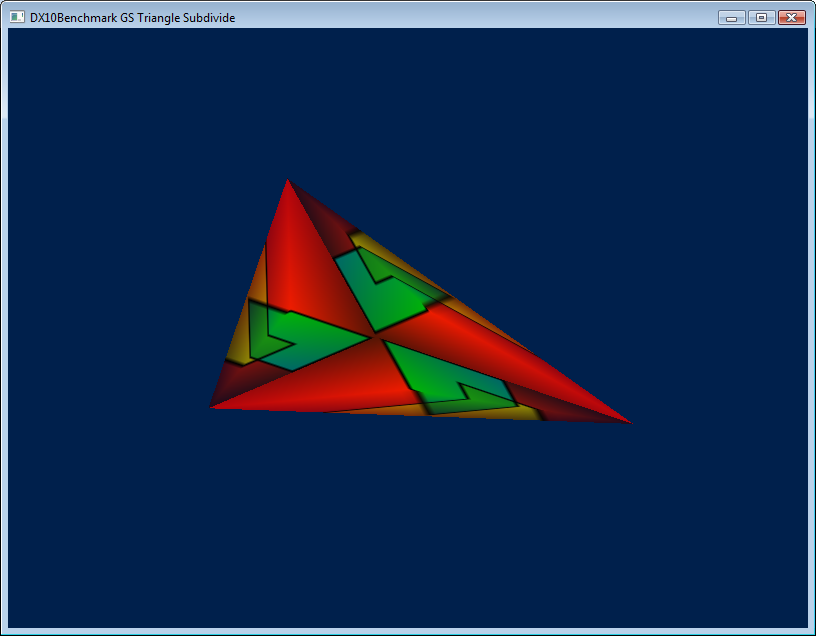


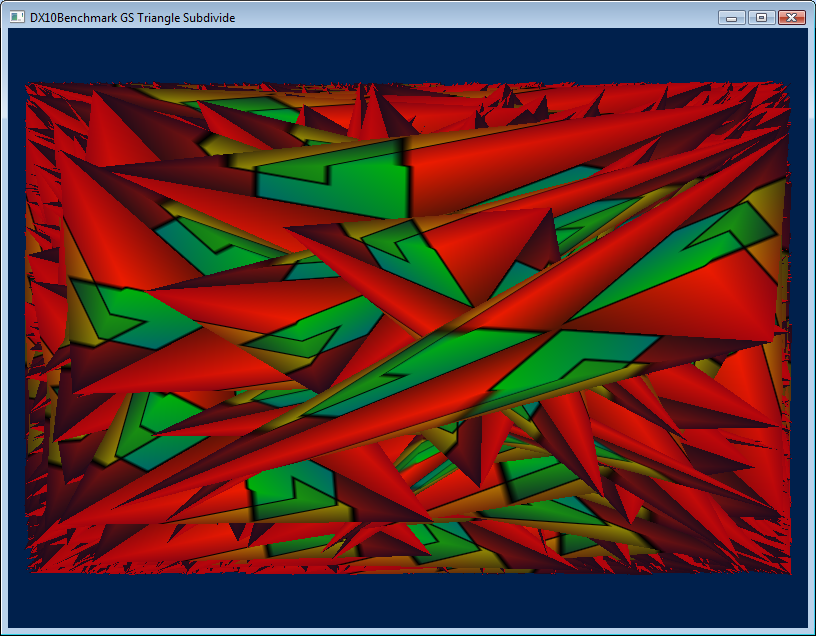
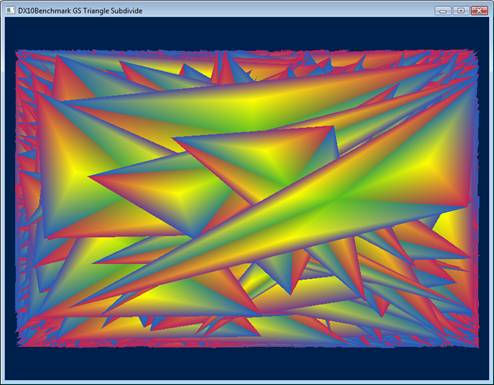
INI: LOOPORCULDISACK=1, POINTSPERBUFFER=3,DRAWCALLSTART=1,DRAWCALLSTEP=0,MAXDRAWCALLS=20000

The second one you see that a pseudo recursive call back to subdivide each of those triangles

Note that by doing a fixed seed for the Random Number Generator I can say without question that the Pixel Shader is doing the same amount of work in both cases.

LOOP with 3 Vertices in the Buffer: 9 Triangles out (27 vertices) (Obviously still Work In Progress)





Actual Interesting Test Cases

