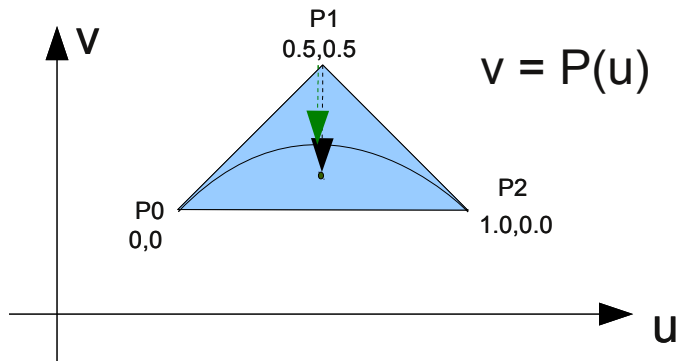


Simple Quadratic, n=2



$$p(u) = \sum_{i=0}^n p_i N_{i,D}(u)$$

where $D=3$, $n=2$

Knot vector = $[0 \ 0 \ 0 \ 1 \ 1 \ 1]$

$v = P(u)$

If interior is required, we change sign of $P1v$
 $\rightarrow P1(0.5, -0.5)$, won't affect the algorithm

Let $A=(u,v)=(u,P(u))$

Let $B=(u, \text{abs}(v)) \rightarrow$ current pixel fragment

if ($v > 0$ and $\|P1B\| > \|P1A\|$)
 or ($v < 0$ and $\|P1B\| < \|P1A\|$)

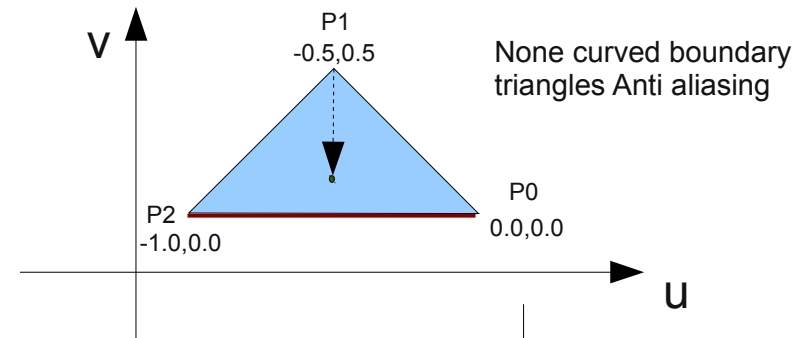
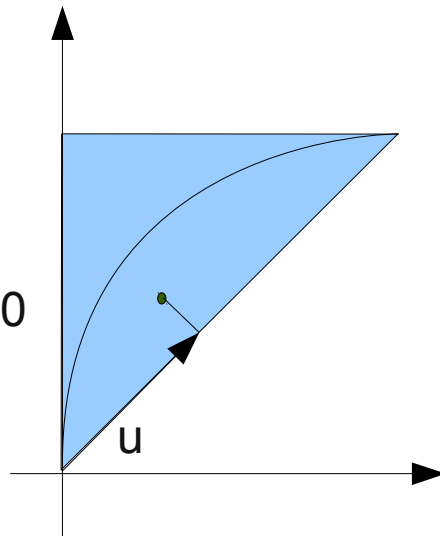
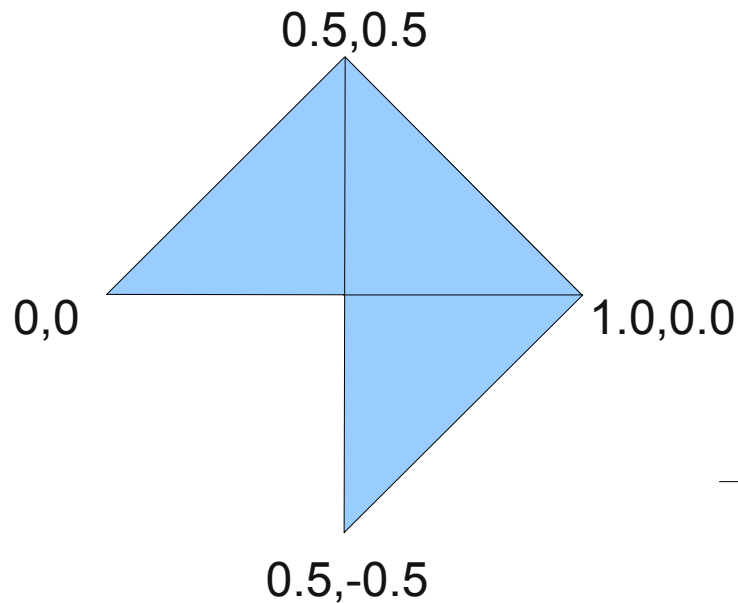
color = c;

c.alpha = func($P1B, P1A$, sign(v))

else

kill

Generalization, $n > 2$



Since $P0P2$ is a boundary edge
 \rightarrow the boundary edge case can be defined
 by signs different of the $P2u$.

Hence

if ($u < 0$)

set color = c;

Let $P1A = (\text{abs}(u), 0)$

$P1B = (\text{abs}(u), v)$

color.alpha = func($P1B, P1A, 0$)

Note: with this addition our
 shader will be able to
 handle the
 boundary triangles and
 Interior triangles.

No need for shader
 switching.