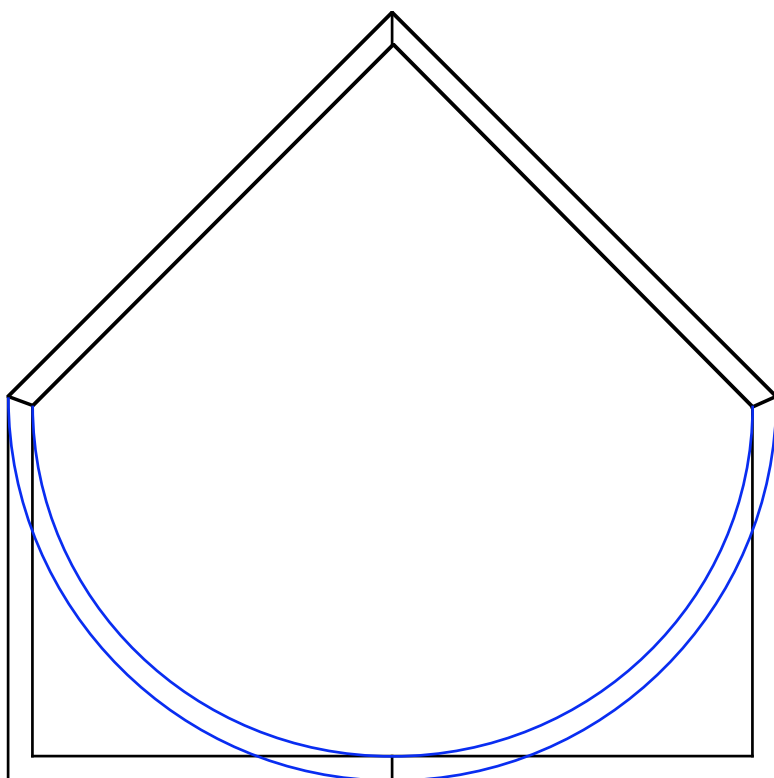


Bevel is created by checking the test point against four-sided polygons that go into the main polygon by a fixed, perpendicular distance.

DIRECTION: For this to work, all characters must be traced in the same direction (clockwise), and all holes must be traced in the opposite direction (counter-clockwise).

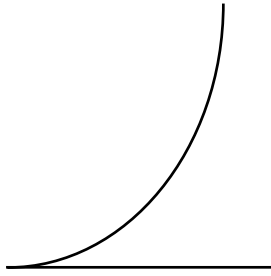
(In the second picture, spline curves are being simulated with circular curves.)



Bevel polygons cannot be constructed just once on app launch, because the bevels need to be uniformly sized across a logo that has differently sized and stretched characters.

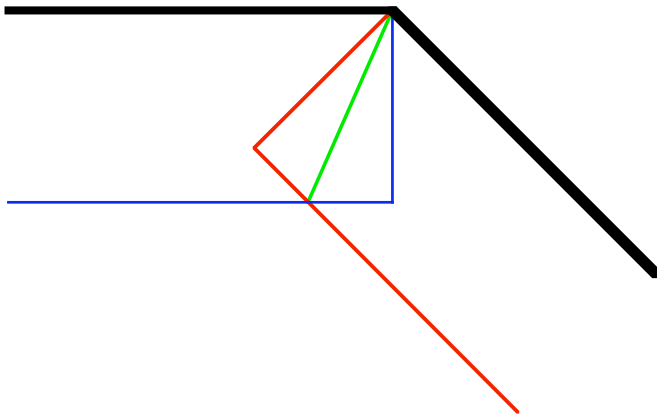
Note that the large, interior area enclosed by the bevel polygons does not need to be constructed as a polygon, because we don't even check the bevel polygons until we've already determined that the test point is inside the enclosing, character shape.

If found to be inside a bevel polygon, the test point should be colored according to the slope at that point.



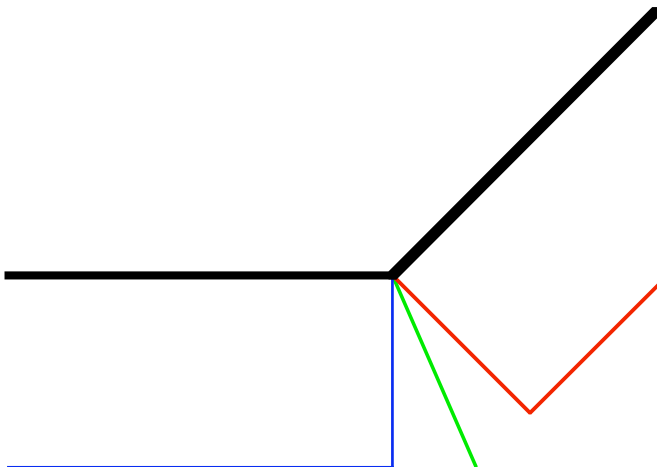
(This does not necessarily need to be related to the side-extrusion vanishing point.)

Third picture: If this situation ever occurs, the bevel corner shoots off to infinity. To avoid this, the bevel algorithm should check for division by zero (or whatever math anomaly results in this situation), and put the bevel corner directly on the polygon's corner — the width of the bevel will constrict to zero there. (Hopefully no character will ever do this!)



Fourth picture: Where two segments of the polygon meet, make parallel copies of each line (blue and red), then meet the bevel areas where the two lines intersect (green).

Fifth picture: Same as fourth picture, but with bevel on outside of angle.



Sixth picture: In theory, when a line segment meets a spline in a non-colinear manner, the spline-point may change significantly (from red dot to pink dot). In practice, this is too complicated, and the bevel is very small — therefore the seventh picture illustrates how it will be done.

