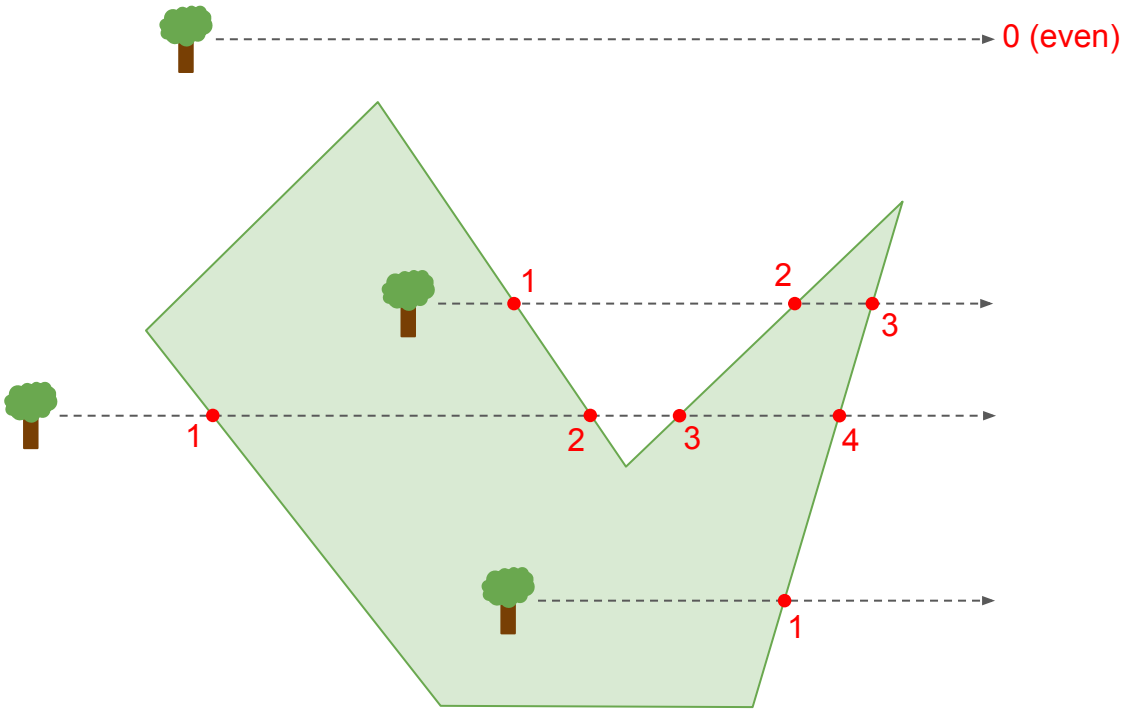
Ray Casting Algorithm (Even-Odd Rule)

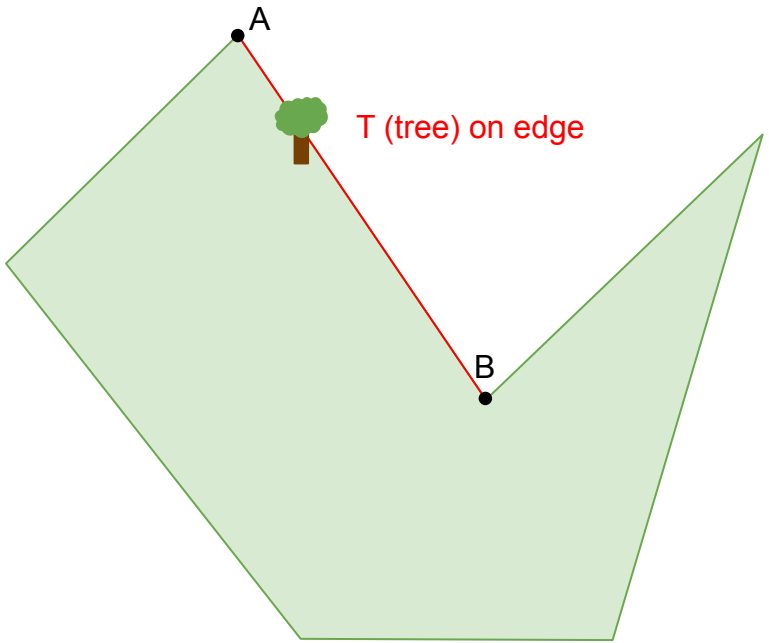


Idea:

1. Cast a horizontal ray rightward from a tree point.
2. Count how many times it intersects with zone edges.
3. Determine relative location based on parity.

Even crossings: Outside
Odd crossings: Inside

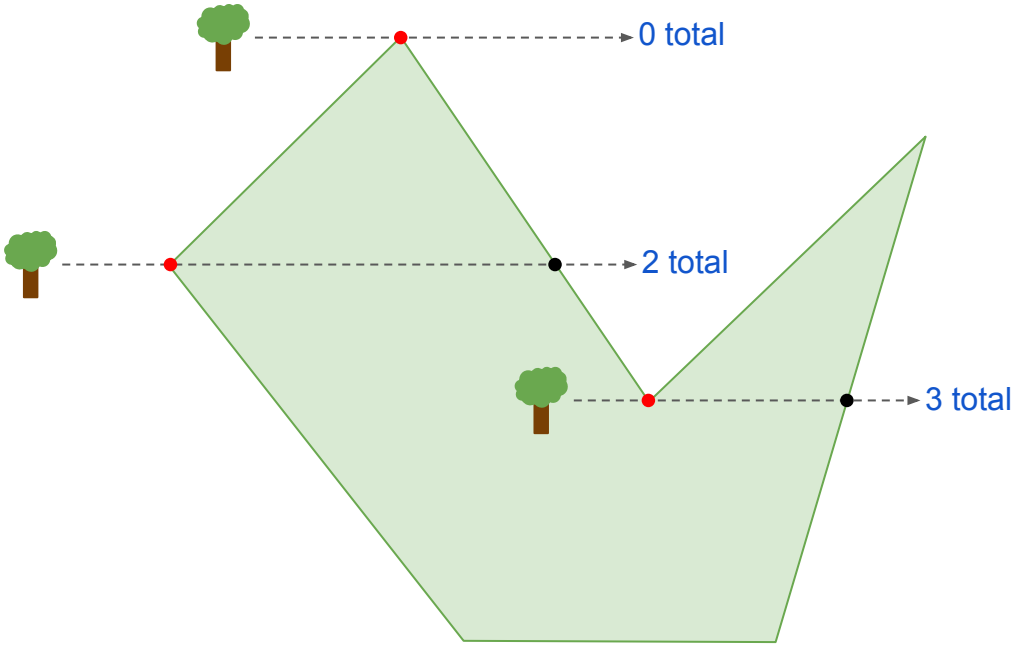
Edge Case: Tree on Boundary



Resolve before ray casting:

1. Find cross product between AT and AB.
2. If cross product is 0, then all 3 points are collinear (on the same line).
3. Verify that T lies between A and B by comparing their coordinates.

Edge Case: Ray Crosses Vertex

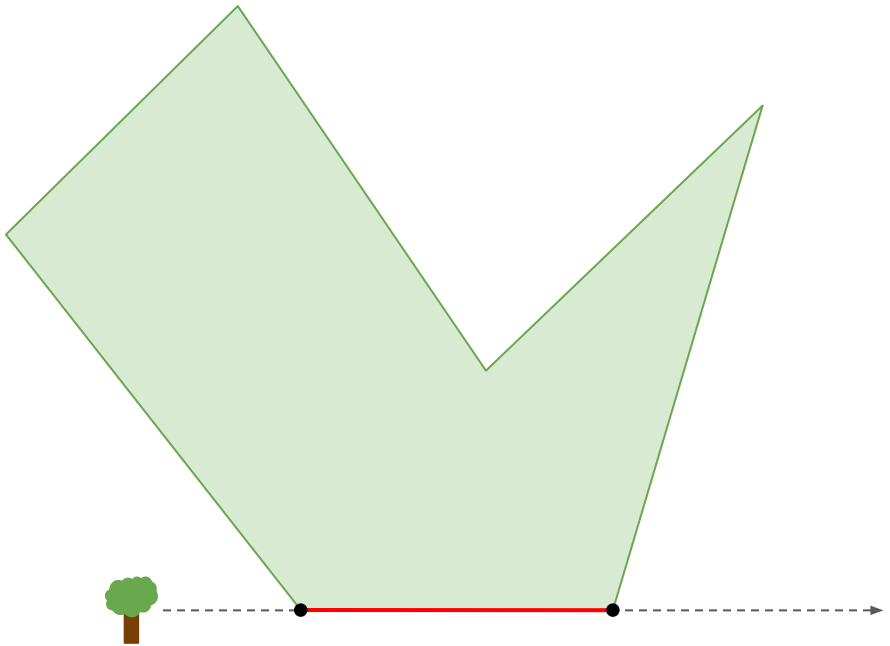


A vertex sits at the meeting of 2 edges. So, how many times we count such crossings?

If we're not careful, that can lead to incorrect location classification.

Solution: Count only crossings with edges that strictly straddle the horizontal ray (For example: one endpoint is strictly above the ray and the other lies at or below it).

Edge Case: Ray Parallel to a Horizontal Edge



This can break the algorithm as it can cause problems such as an invalid crossings calculation or disrupting parity logic.

It can be detected by checking if the difference between endpoints is 0.

Solution: Ignore horizontal edges as they don't represent true boundary crossings.