```
fStepDistance
      The path is broken up into small steps of this length.
fLookAheadDistance > fStepDistance
      How far along to path to look ahead.
```

The desired rotation is the direction from the current location to the
first step on the path that is further away than fLookAheadDistance.
The path may be broken up into fixed sized steps so we can loop
and compare against distance squared, instead of calculating it exactly
using lots of trig functions.

```
fSnapTurnAngle0 < fSnapTurnAngle1 < fSnapTurnAngle2 <
fAnimateTurnAngle1 < fAnimateTurnAngle2 < fAnimateTurnAngle3
      The cut off angles for different turn speeds.
```

If the absolute value of the change in rotation is less than
fSnapTurnAngle[0, 1, 2], then snap to the desired rotation,
and walk with [full, half, quarter] steps.
If it's less than fAnimateTurnAngle[0, 1, 2], then turn by animating
the [45, 90, 180] degree turn, twisting it to adjust the
animation's rotation to the desired final direction.