**Corner Points Extraction Algorithm**

1. All points in each scanline are separated into clusters Ci using threshold check on theta & range differences. Discard those clusters whose number of points are less than a threshold.
2. For jth point in the ith cluster Ci, select neighboring points Nj, using sliding window method whose size is s. S is an odd integer.
3. Compute geometric features for jthpoint, where j is in {1, …., ni}. ni is total points in cluster Ci.
4. For jth point, dot product is computed by splitting Nj into two halves of points: preceding half of points Pj and succeeding half of points Sj
5. PCA is computed on Pj & Sj. 1ej & 2ej and 1fj & 2fj are first two singular values. Similarly, 1uj and 1vj are first singular vectors of Pj and Sj.
6. Single dimensionality check: If ratios mj = 1ej / 2ej and nj = 1fj / 2fj are greater than Ƭsv = 10, assign Pj and Sj as linear otherwise not linear. Ƭsv is threshold for ratio of singular values.
7. Compute dot product feature for the jth point as DPj = <1uj,1vj>, if both Pj and Sj are linear.
8. Compute angle between preceding & succeeding points for jth point as α = cos -1(DPj/ (||1uj || ||1vj||)).
9. If jth point is selected as corner point if DPj <= ƬDP. ƬDP = 0.6 is threshold value for dot product feature.

**Requirement:**

Minimum requirement: Output index of set of points containing corner without any false detection.

Preferred requirement: Locate the exact corner point in each of corner containing set of points.

Summary

* Write algorithm to pin point on the corner points
* Decide on the threshold for cluster separation & outlier designation of cluster
* Write c++ codes : function for each step in algorithm