Efficiency Analysis of Quasi-Random Sampling Algorithm for Contents-Based Image Retrieval

Jyun-Hao Huang

Supervisors: Dr. Jun Zhou
Dr. Antonio Robles-Kelly

Content Based Image Retrieval

- Query By Image Content

Content Based Image Retrieval

- Preprocessing

Content Based Image Retrieval

- Query

Algorithms

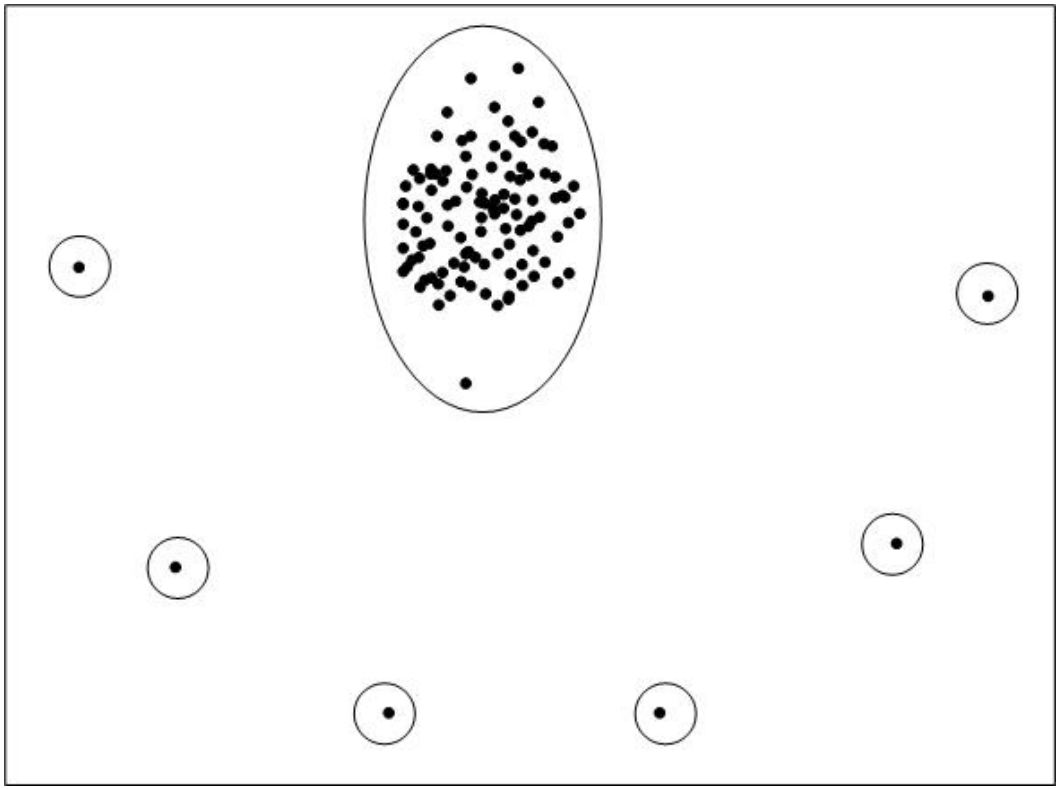
- Quasi-Random Sampling Algorithm (QuaRS)
- K-Dimensional Tree (KD-Tree)

QuaRS

- Clustering
- Image Retrieval

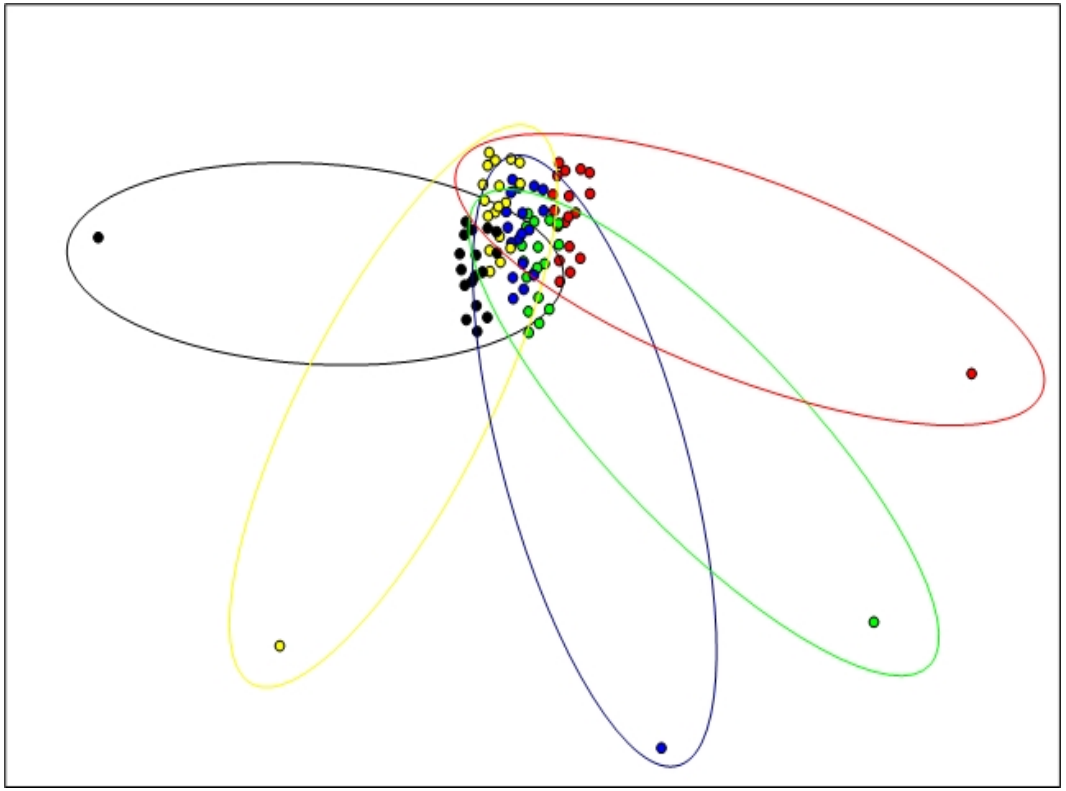
QuaRS

- Clustering K-Means



QuaRS

- Clustering　Expectation Maximization(EM)



QuaRS

- Clustering

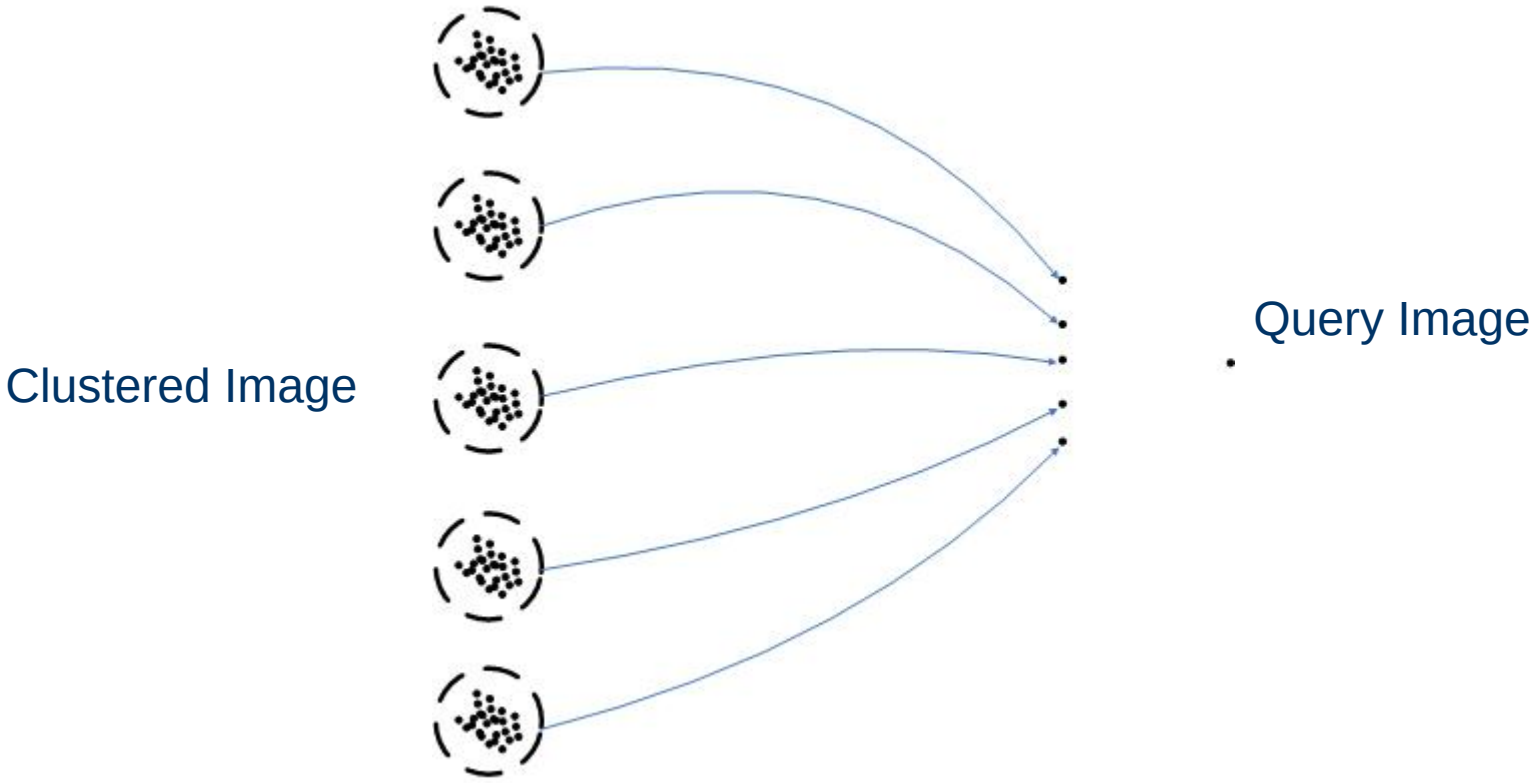
Given N images, we want

$\sqrt{N}$ Clusters &

$\sqrt{N}$ images in each cluster

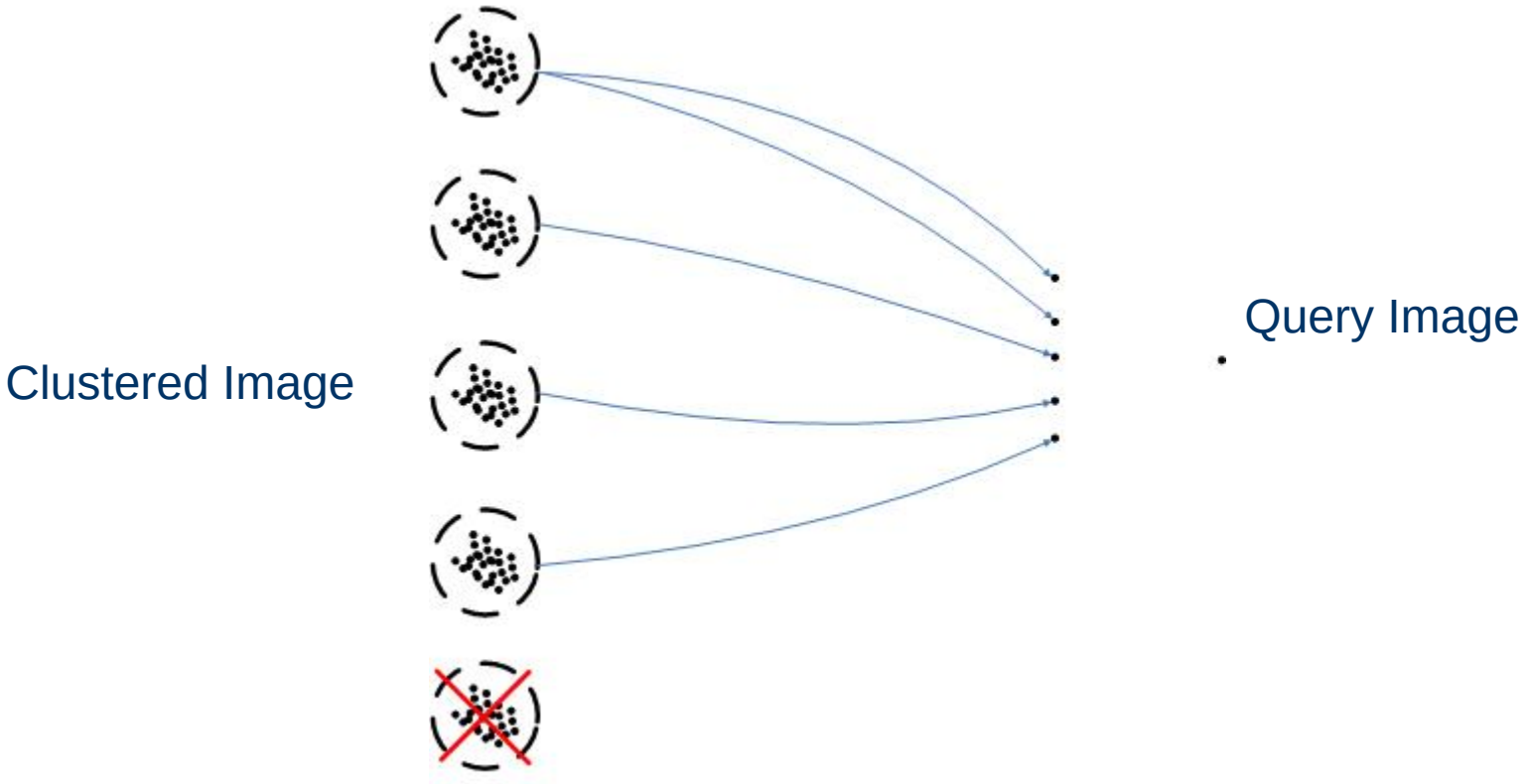
QuaRS

- Image Retrieval



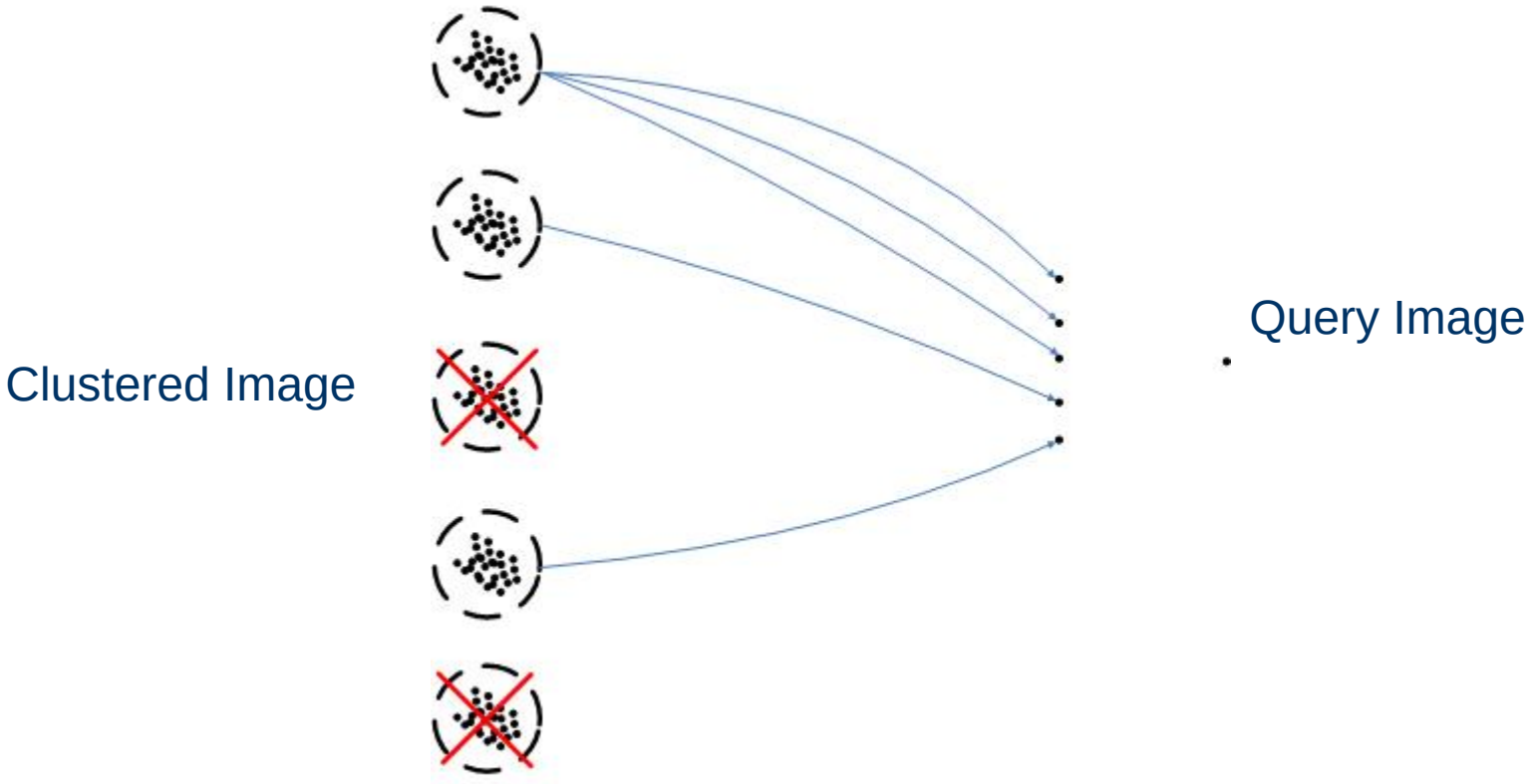
QuaRS

- Image Retrieval



QuaRS

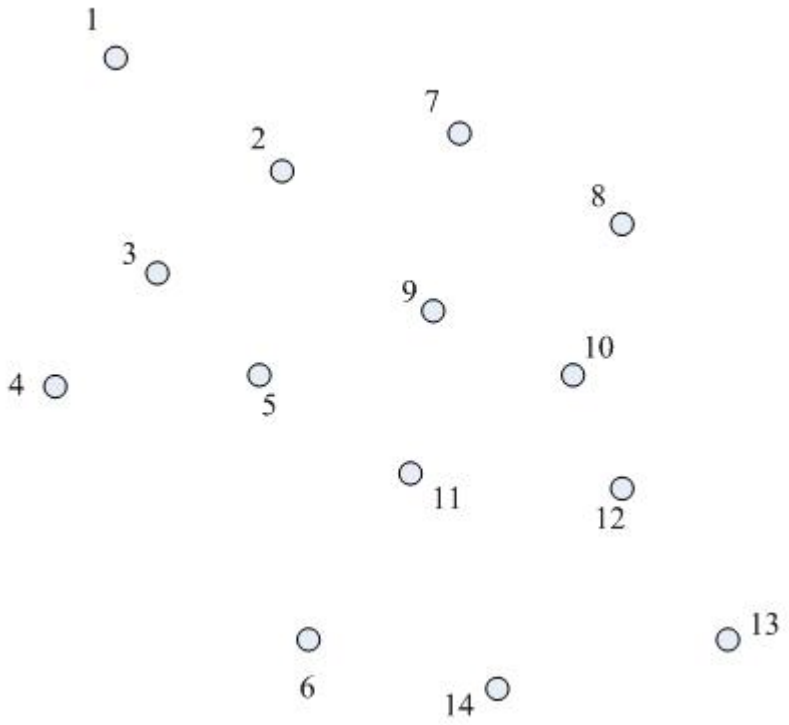
- Image Retrieval



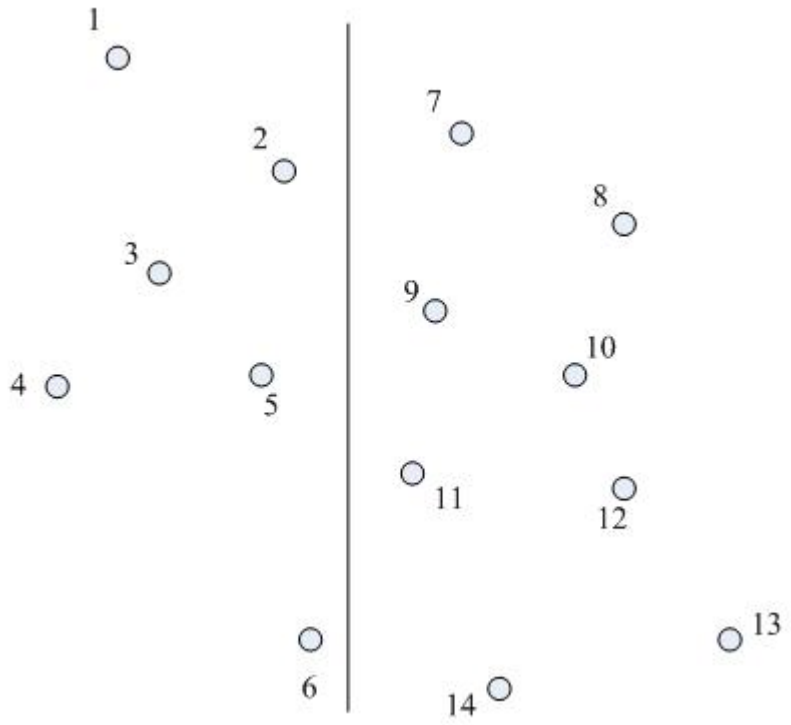
KD-Tree

- A Binary Tree
- Basic algorithm:
  1. Select a dimension as the axis to split the data.
  2. Find the median value of the axis across the whole dataset.
  3. Create an new node.
  4. Split the dataset into two subsets using the axis and the median value found.
  5. For each subset, repeat step 2 to 4 using other dimensions.

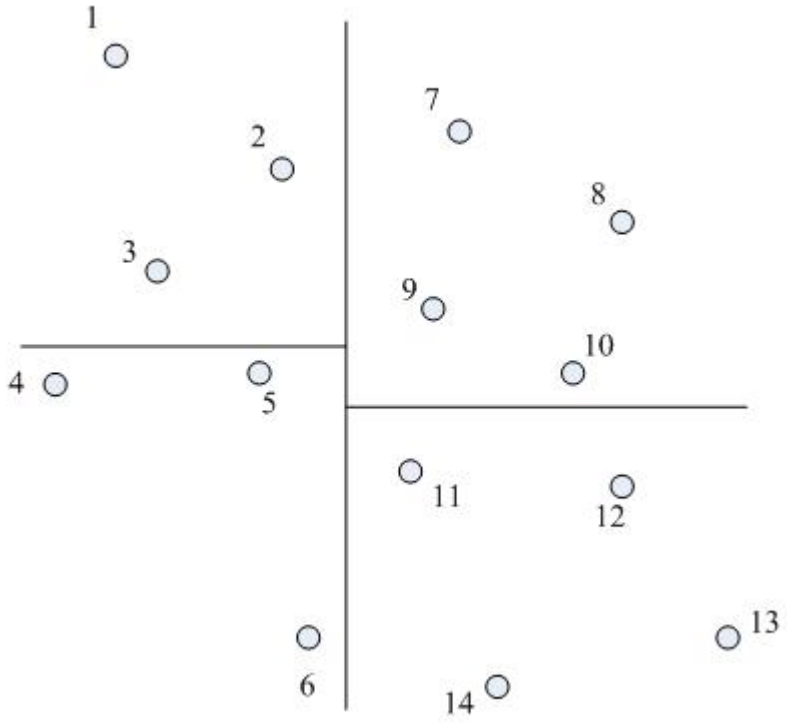
KD-Tree



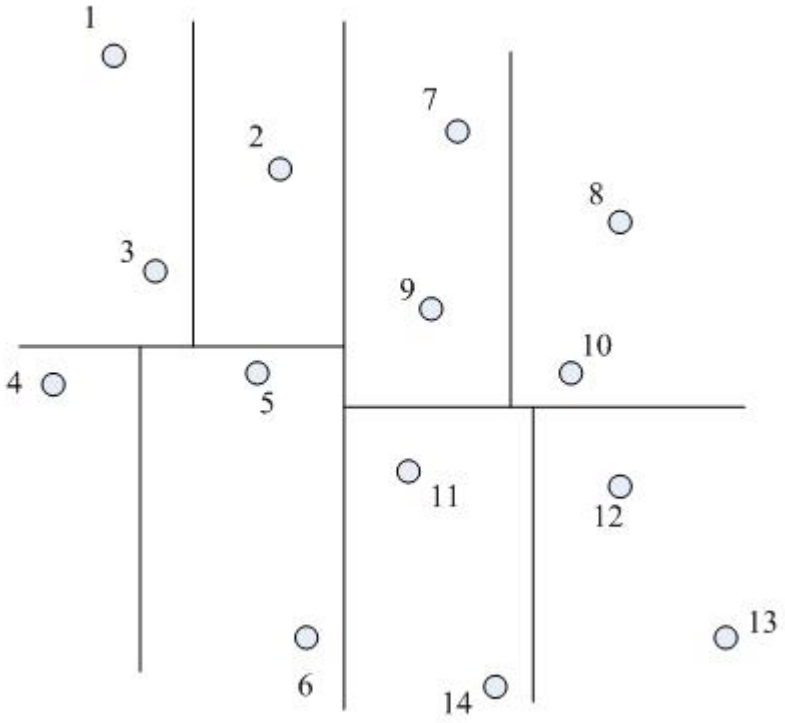
KD-Tree



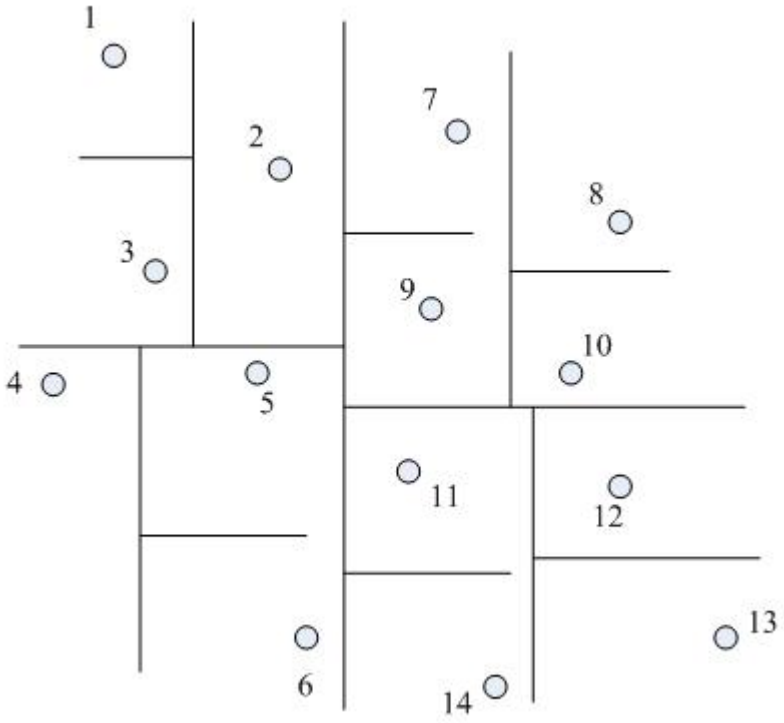
KD-Tree



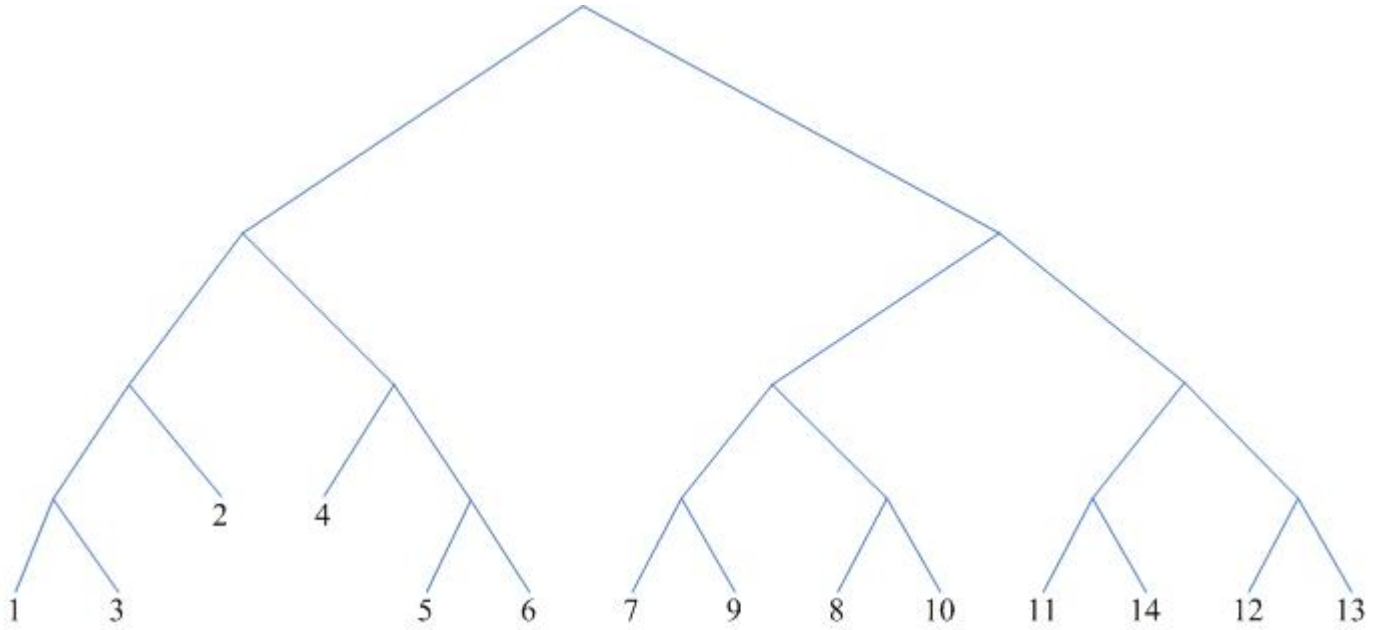
KD-Tree



KD-Tree



KD-Tree



Experiment

- Implemented using Standard C++
- Boost C++ Library (version 1.36.0)
- Features:
  - Color
  - SIFT (Scale-invariant feature transform)
  - Texture
  - Concatenated
- Dataset: Caltech 101

Experiment

- K Nearest Neighbor (K-NN)

- Accuracy measurement

  accuracy = (total hits) / (total queries)

Experiment

- Intensive V.S. QuaRS

	1-NN	2-NN	3-NN	4-NN	5-NN	Sampling
Intensive	23.6 %	35.3 %	40.3 %	46.0 %	49.7 %	100 %
QuaRS	20.6 %	27.3 %	33.0 %	36.7 %	40.3 %	35.7 %

Experiment

- QuaRS under different dataset scales

	1-NN	2-NN	3-NN	4-NN	5-NN	Sampling
20	20.6 %	27.3 %	33.0 %	36.7 %	40.3 %	35.7 %
30	12.2 %	17.3 %	21.3 %	24.2 %	28.0 %	31.5 %
40	13.5 %	17.6 %	21.6 %	24.8 %	27.8 %	29.0 %
50	12.0 %	15.6 %	18.8 %	21.0 %	23.0 %	28.6 %

Experiment

- QuaRS V.S. KD-Tree (Accuracy)

	1-NN	2-NN	3-NN	4-NN	5-NN	Time
QuaRS	12.0 %	15.6 %	18.8 %	21.1 %	23.1 %	619 s
KD-Tree	7.2 %	10.3 %	12.3 %	13.9 %	16.4 %	359 ms

Experiment

- Boost is not optimized !!
- prod(matrix1, matrix2)
- Other matrix operations are not efficient too !!

Conclusion & Future Works

- Replace Boost
- Combine Tree structure with QuaRS