K-D TREE

Master's of Computer Application

April 27, 2023

MNNIT
ALLAHABAD

TEAM MEMBERS

-

1. 2022CA013 Arnab Kar
2. 2022CA021 Bikash Sah
3. 2022CA036 Himanshu Negi
4. 2022CA043 Kartik Chandra
5. 2022CA064 Prakhar Solanki
6. 2022CA077 Raunak Kumar
7. 2022CA081 Rishiraj Patel
8. 2022CA096 Saurabh Yadav
9. 2022CA108 Suthar Dinesh

Outline of the Presentation

- Introduction
- Foundation of K-D tree
- Construction of K-D tree
- Querying K-D Tree
- Applications of K-D tree
- Advantages
- Disadvantages
- Conclusion
- References

Introduction

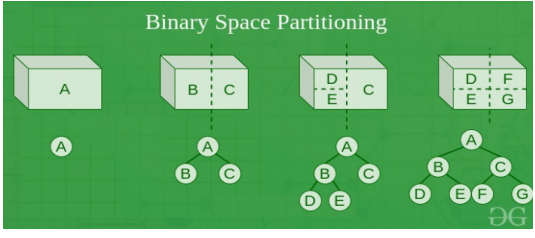
- In computer science, a KD tree is a binary tree data structure used for partitioning a multi-dimensional space into smaller regions for efficient search and data analysis.
- It was invented by Jon Louis Bentley in 1975.
- Based on binary search tree concept.
- Very useful for range and nearest neighbor searches.
- Data in each node of KD Tree is a K-Dimensional information in space.

Foundation of K-D Tree

o

Binary Space Partitioning

- BSP (Binary Space Partitioning) is a general technique for recursively dividing space into two parts using a hyperplane.
- Each node in the tree represents a subspace of the overall space, with the root node representing the entire space.



Spatial Subdivision

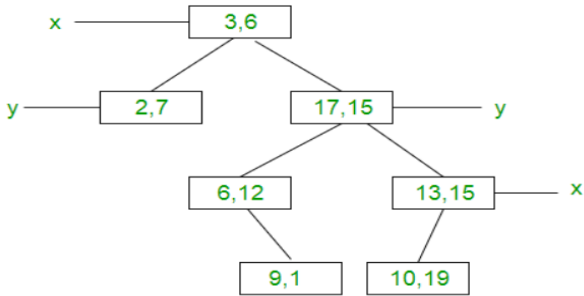
- The process of dividing the space into smaller regions is known as spatial subdivision.
- In KD Tree, the space is divided into smaller regions by hyperplanes that are perpendicular to the coordinate axes. This process continues until each region contains only a single point, at which point the tree is fully constructed.

Construction of K-D Tree

- Make first node as root
- While insertion, first search with X then choose left and right.
- If the left or right empty, add node accordingly.
- If left or right not empty then search with Y then choose left or right.
- Do you search alternate on X and Y value and find empty left or right.
- Repeat these steps recursively.

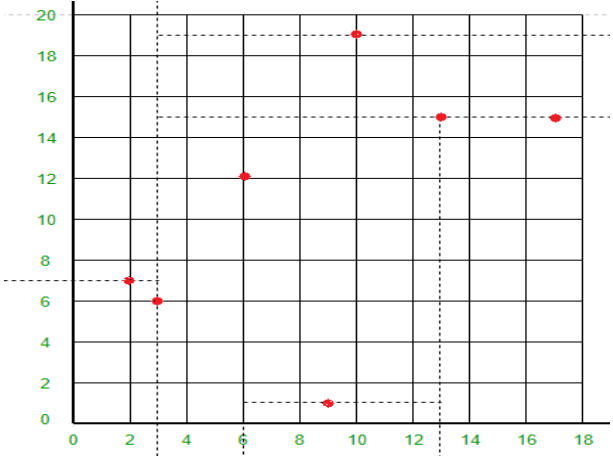
Construction of K-D Tree

- Example: (3, 6), (17, 15), (13, 15), (6, 12), (9, 1), (2, 7), (10, 19)



Construction of K-D Tree

- Example: (3, 6), (17, 15), (13, 15), (6, 12), (9, 1), (2, 7), (10, 19)



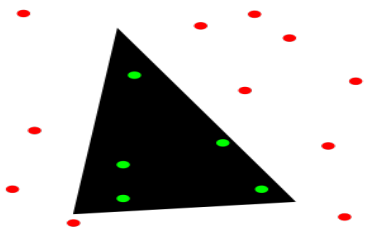
Querying K-D tree

- **Nearest Neighbor Search**
  - Find the point in the KD tree that is closest to a query point.
  - Traverse the tree and compare the query point to the median point of each node.
  - Update the best-so-far point as the search proceeds.
  - Backtrack to parent nodes to check whether the other sub-tree needs to be searched.

Querying K-D tree

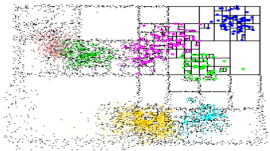
o

- **Range Search**
  - Find all points in the KD tree that are within a radius of a query point.
  - Traverse the tree and check whether the search radius intersects the splitting hyperplane of each node.
  - Add the point to the result set if it is within distance r of q.

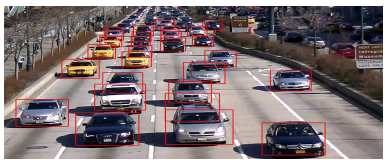


Applications Of K-D Tree

- **Image Processing :** KD tree are used in image processing applications . It can be used to search for similar images based on their pixel values or features.



- **Computer Vision :** KD tree is used in computer vision applications such as object recognition .



Application of K-D tree

- **Machine Learning :** One of the most common uses is nearest neighbor search, where they're used to find the closest point to a given point quickly.

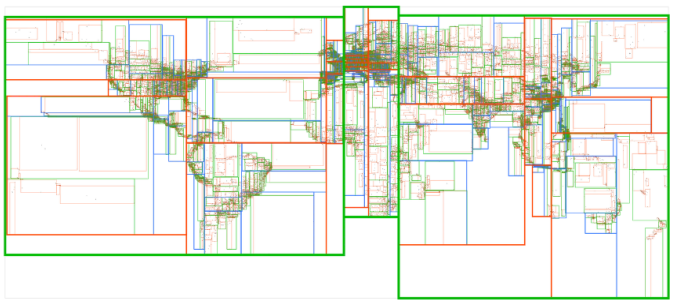
(a) Core and Delta

(b) Fingerprint structure

- **Robotics:** KD tree have been used in robotics for path planning and obstacle avoidance.

Application of KD tree

- **Geographic Information Systems:** KD trees are used in geographic information systems (GIS) for geometrical data analysis and visualization. It can be used to search for points of interest in a geographic region based on their coordinates.



Advantages

o

- **Fast nearest neighbor search:** KD tree can quickly find the nearest neighbor of a query point in high-dimensional space.
- **Memory efficiency:** KD tree uses a tree structure to organize data points, which saves memory compared to other nearest neighbor search algorithms that require storing all data points.
- **Efficient for low-dimensional data:** KD tree is very efficient for low-dimensional data, where the number of dimensions is relatively small.

Disadvantages

- **Sensitive to input order:** The efficiency of kd-trees is highly sensitive to the order of the input data. If the data is poorly distributed, the tree's balance can be compromised, leading to inefficient search times.
- **High-dimensional curse:** While kd-trees are efficient in high-dimensional spaces for nearest neighbor search, they are still subject to the curse of high dimensionality, where the search time complexity grows exponentially with the number of dimensions.
- **Limited to static data:** Kd-trees are suitable for static data that does not change frequently. Insertion or deletion of data from a kd-tree can be challenging, and may require rebuilding the entire tree, which can be expensive.

Conclusion

- In conclusion, the KD tree is a versatile and efficient data structure for organizing and searching multidimensional data points. It has been widely used in various fields such as spatial data analysis, image and video processing, machine learning, computer graphics, bio-informatics, and IoT applications.
- The KD tree ability to efficiently partition space and perform range searches, nearest neighbor searches make it a valuable tool for speeding up algorithms and processing large datasets.
- Despite its advantages, the KD tree also has limitations, such as sensitivity to data distribution,Limited to static data.

References

- For LaTex: Overleaf Framework
- Wikipedia , YouTube contents .
- Blum, M., Floyd, R.W., Pratt, V., Rivest, R.L., and Tarjan, R.E. Time bounds for selection. Stanford CS Rep.
- Geeks For Geeks , Ashish Barnwal article.
- And other internet resources.

K-D TREE

Thank you for listening!

Any questions?