

---

# Nearest Neighbor Search Data Structure

---

**Sanjoy Dasgupta**

Department of Computer Science  
University of California, San Diego  
dasgupta@cs.ucsd.edu

**Zhen Zhai**

Department of Computer Science  
University of California, San Diego  
zzhai@ucsd.edu

**Eugene Che**

Department of Computer Science  
University of California, San Diego  
eche@ucsd.edu

## Abstract

Nearest-neighbor(NN) search is boardly used within all different fields of study to gain information on new data from training data set. For NN search, the more complex the training data set is the more accurate the result will be. However, doing NN search on complex and large training data sets is time consuming. Therefore, improving the speed and accuracy of nearest-neighbor search becomes essential. We look at different data structures for NN search and compare the results on varied data sets. We focus on data structures including KD tree and PCA tree. We also look at KD spill tree, KD virtual spill tree, and PCA spill tree to explore about the technique of spilling and virtual spilling. We conclude that spilling and virtual spilling improve the performance of the data structures.

## 1 Introduction

Gaining information of an input query by looking for the most similar match from the training set is referred as nearest neighbor(NN) search. We are given a data set  $\mathbf{S}$  that contains a set of points  $\{s_1, s_2, s_3, \dots, s_n\}$  and an input query  $q$ , and we are looking for the nearest neighbor  $s_i$  of  $q$  in  $\mathbf{S}$  such that  $s_i$  can provide valuable information about input query  $q$ .

A lot of machine learning algorithms spend a big amount of time on nearest neighbor search. The reason that NN search is time consuming is that the training set could very well be represented by high dimensional vectors. In fact, the accuracy of the NN search result is proportional with the complexity of the training set. Therefore, speeding up NN search can lead to tremendous improvement to many applications of not only machine learning but also data analysis, bio-informatics, signal processing, etc.

There has been a large number of research devoting to speed up the performance of NN search. In this paper, we evaluate two most promising data structures, KD tree and PCA tree. We conclude that PCA tree has better performance than KD tree.

We also experiment the technique of spilling by looking at KD spill tree vs. KD tree and PCA spill tree vs. PCA tree. We observe that spilling gives a better result.

We finally experiment the technique of virtual spilling by looking at KD virtual spill tree. We compare it with KD spill tree and conclude that virtual spilling improve the performance of spilling.

We apply these data structures on five different data sets from different area of study including computer vision, human relationship... We focus on the accuracy of NN search using these data structures and make sure it generally apply to all different kinds of data sets. In conclusion, we observe the following: PCA tree has a better performance comparing to KD tree, spilling improve the performance, and virtual spilling improve the accuracy slightly more than spilling.

## 2 KD tree vs. PCA tree

We focus our study on binary space partitioning(BSP) trees, one of the most promising data structure for NN search. We first look at k-dimensional(KD) trees. Then we study principle component analysis(PCA) tree, in which we preprocess the data using PCA(Pearson, 1901) before partitioning the space.

## 2.1 KD trees

KD tree is a BSP tree that partition data points into a  $k$  dimensional binary tree (Bentley, 1975). Starting from the root of the tree, a coordinate with biggest variance is chosen. The data points  $S$  from root are split at the median of the coordinate into two children of the root. Recursively apply the process on the children until the leaf of the tree contain at most  $n$  data points.

Given KD tree  $T$  and query  $q$ , simply trace  $q$  down the tree and find the leaf  $l$  it falls in. Return all the possible nearest neighbors within leaf  $l$ . Then do distance calculation for each of the data point in leaf and return the point that lies closest to query  $q$ .

Pseudocode:

```
function BuildTree(S, n)
  If  $|S| < n$ 
    return S as leaf
  Choose the coordinate  $c$  with biggest variance
  Split S into L and R at the median  $m$  of  $c$ 
  Record coordinate  $c$  and mean  $m$ 
   $LeftTree = BuildTree(L, n)$ 
   $RightTree = BuildTree(R, n)$ 
function SearchTree(T, q)
  If T is leaf
    return T
  Retrieve  $c$  and  $m$  from T
  If  $q[c] < m$ 
     $SearchTree(L, q)$ 
  Else
     $SearchTree(R, q)$ 
```

The problem with KD tree is that the true nearest neighbor  $n$  of query  $q$  could be in a leaf that is different from the leaf  $q$  falls in. This could happen easily when  $q$  and  $n$  are both located close to the boundary of the tree (Figure 1). Therefore, the error rate could be very high.

## 2.2 PCA tree

PCA tree is also a BSP tree which partition data after a process of PCA on the data points. From the root of the tree, we do a PCA and find the first principle component  $v$ . We then reduce the dimension of the data by mapping all the data points onto  $v$  which gives us a one dimensional data set  $s$ . The original data points from root are partition using the corresponding data in  $s$  and split at the median of  $s$ .

Similar to KD tree, searching NN of a query  $q$  can be done by tracing  $q$  down the tree.

Pseudocode:

```
function BuildTree(S, n)
  If  $|S| < n$ 
    return S as leaf
  Run PCA and find the first principle component  $c$ 
  Map S onto  $c$  to get  $S_c$ 
  Split  $p$  at the median  $m$  of  $S_c$ 
  Split S according to  $p$  into L and R
  Record principle component  $c$  and mean  $m$ 
   $BuildTree(L, n)$ 
   $BuildTree(R, n)$ 
function SearchTree(T, q)
  If T is leaf
    return T
  Retrieve  $c$  and  $m$  from T
```

```
Map  $q$  onto  $c$  to get  $q_c$   
If  $q_c < m$   
   $SearchTree(L, q)$   
Else  
   $SearchTree(R, q)$ 
```

## **2.3 Experimental results**

# **3 Spilling or not**

We explore the technique of spilling on both KD tree and PCA tree.

## **3.1 Spill tree**

Spill tree is ...

## **3.2 Experimental results**

# **4 Virtual spill tree vs. spill tree**

## **4.1 Virtual spill tree**

Virtual spill tree is ...

## **4.2 Experimental results**

# **5 Conclusion**