

Student's project

RYDZ-WULLENS, ROUX

4 décembre 2025

# 1 Introduction

Light Detection and Ranging (LiDAR) is a technique for measuring distances using pulses of light. Its built upon the principle that when a light pulse is directed towards a certain target, the distance to the target is proportional to the time it takes for pulse to reach the target, reflect and return to the emitter. By scanning a 3D structure with these light pulses, a point cloud representing the surface can be constructed. However, LiDAR data is often noisy and may not fully capture the underlying surface, which makes further processing necessary.

One key processing step is finding the nearest neighbors of each point in the cloud. This information allows the local surface geometry to be estimated, enabling the computation of surface normals and the application of smoothing or filtering techniques. Performing a brute-force search for nearest neighbors in a point cloud containing millions or even billions of points comes at an enormous computational cost. Thus, efficient data structures such as KD-trees are commonly used to greatly reduced this cost.

## 2 Kd-trees

### 2.1 Definition

Let  $K = (V, E)$  be a binary tree. Let  $v$  be a node of  $K$ ,  $\text{depth}(v)$  denote the depth of  $v$  in  $K$ . Let  $\text{anc}(v) \subset V$  the set of the ancestors of  $v$  in the tree. Let  $l_c : v \mapsto l_c(v)$  and  $r_c : v \mapsto r_c(v)$  the functions that maps a node  $v$  to its left (resp. right) child in the tree. Let  $\text{val} : (v, i) \mapsto A_v(i)$  the fonction that maps  $(v, i)$  to the value stored in  $v$  along the  $i$ -th dimension. Let  $\text{lst} : v \mapsto \{w \in V | l_c(v) \in \text{anc}(w)\}$  and  $\text{rst} : v \mapsto \{w \in V | r_c(v) \in \text{anc}(w)\}$

A KD-tree is defined by the following property.

$$\forall v \in V, \forall w \in \text{anc}(v)$$

$$\text{val}(v, \text{depth}(w) \bmod k) > \text{val}(w, \text{depth}(w) \bmod d) \text{ if } \exists w' \in \text{anc}(v) \bigcup \{v\} \text{ such that } r_c(w) = w'$$

$$\text{val}(v, \text{depth}(w) \bmod k) < \text{val}(w, \text{depth}(w) \bmod d) \text{ if } \exists w' \in \text{anc}(v) \bigcup \{v\} \text{ such that } l_c(w) = w'$$

### 2.2 Properties

$\forall v, v' \in V, v \neq v'$  let  $w^*$  denote the closest common ancestor.  $w^* \in \text{anc}(v) \cap \text{anc}(v')$ ,  $\forall w \in \text{anc}(v) \cap \text{anc}(v') : \text{depth}(w^*) \geq \text{depth}(w)$

We have the following :

$$\text{val}(v, \text{depth}(w^*) \bmod k) - \text{val}(v', \text{depth}(w^*) \bmod k) > 0 \text{ if } v \in \text{rst}(w^*)$$

$$\text{val}(v, \text{depth}(w^*) \bmod k) - \text{val}(v', \text{depth}(w^*) \bmod k) < 0 \text{ if } v \in \text{lst}(w^*)$$

### 2.3 Motivation

Kd-trees are a special type of Spatial Data Structure along with spatial hash-tables or K-trees. They allow to efficiently search the nearest neighbor of a point  $q$  by going through a binary tree. If the Kd-tree is well balanced, which

can be guaranteed using auto-balancing or by inserting the points in a certain order, the time complexity of finding the nearest neighbor is  $O(\log N)$ . In fact, searching such a point is going through the BST.

#### **2.4 Finding the nearest neighbor**