CSI2110 Programming Assignment 2 Report

Matthew Petrucci

300119235

# **Introduction**

This programming assignment is a continuation of the first programming assignment, where clusters are given that represent points in 3D space that reflect a given environment. We must use data structures we make to cluster these points together depending on their distance to one another and how many points are required to be in a given distance to be considered a cluster. The first assignment used a linear stack as the algorithm to classify the points, while assignment 2 aims to improve the run-time of the stack by implementing a KD tree to hopefully reach a minimum of O(log(n)) time, instead of the linear stack O(n) time.

The data shown in this assignment report will almost always be represented as a contrast between both the linear stack implementation and the KD tree implementation, with the exception of Experiment 3 which does not ask for a comparison between KD and linear times. All data can be seen in their raw format in their respective folders.

# Experiment 1

Experiment 1 consisted of us given several 3D points and asked to find all neighbours and the amount of neighbours using both a given (and correct) linear stack implementation and our own KD tree implementation. This experiment was mostly to verify that the KD tree implementation worked, and the tests prove tat, although both methods work differently, they provide the same results. Here is a short exerpt from the folder.

(-5.429850155, 0.807567048, -0.398216823):

number of neighbors = 5

[(-5.420458778974271,0.7891803562243134,-0.3973486218703048), (-5.429850154613408,0.8075670478362598,-0.3982168226988382), (-5.43030556398262,0.8246710769927127,-0.3984338736632657), (-5.432677820578597,0.8420909833742529,-0.3987956432309413), (-5.415942549526783,0.7715622302147948,-0.3968421613600826)]

(12.97637373, 5.09061138, 0.762238889)

number of neighbors = 2

[(-12.992860583393504,5.051138148093654,0.7622934861842156), (-12.976373725118926,5.090611379773172,0.7622388885867976)]

javac \*.java && java Exp1 kd 0.05 Point\_Cloud1.csv -36.10818686 14.2416184 4.293473762

number of neighbors = 1

[(-36.10818686248445,14.241618397722052,4.293473761897471)]

javac \*.java && java Exp1 kd 0.05 Point\_Cloud1.csv 3.107437007 0.032869335 0.428397562

number of neighbors = 17

[(3.106996584743287,0.0607965822112821,0.3982019007320806), (3.1121272286089696,0.0513459811348819,0.3988373764321324), (3.123179921500925,0.041944387155322,0.4002354917039907), (3.1233786019830654,0.0227801237087123,0.4002354923207727), (3.1274271451816897,0.0036195964459265,0.4007438901352219), (3.1431306038656457,0.032567946655484,0.4027774803812885), (3.124626229458665,0.071407284130788,0.4308557147843563), (3.101060042170619,0.0613504651643957,0.4275782041593502), (3.102224266499409,0.0518528106206318,0.4277147594345744), (3.1074370070938127,0.0328693350571258,0.4283975616247083), (3.110494894440637,0.0233580135464159,0.4288072530996791), (3.124448583809515,0.0042909825235033,0.4307191323087426), (3.127132326652067,0.0426729987558792,0.4311288249443483), (3.1491863135474705,0.0139866281830143,0.4341331975609065), (3.120500565609992,0.0626817334571532,0.4610553779823442), (3.1325477563686954,0.0533098761028178,0.4628090065256689), (3.135663991597881,0.0437404978846847,0.4632473987090112)]

javac \*.java && java Exp1 kd 0.05 Point\_Cloud1.csv 11.58047393 2.990601868 1.865463342

number of neighbors = 2

[(11.597053489523276,3.032865894391464,1.8696242228185609), (11.580473933555549,2.9906018684790574,1.8654633424019456)]

javac \*.java && java Exp1 kd 0.05 Point\_Cloud1.csv 14.15982089 4.680702457 -0.133791584

number of neighbors = 2

[(14.159820885717384,4.680702456874969,-0.1337915844837233), (14.180766680737111,4.639415392714257,-0.1338543779659772)]