1. Stempoint.m: slice processing and Euclidean distance clustering to obtain candidate points

Input: original data 1.ply files.

Output: the current tree stem points and we save it in 1\_stempoints.txt.

1. Remove\_dup.m: remove redundant points or spatial close points. (this step is not necessary)

Input: candidate points in stempoints.txt.

Output: the three-dimensional coordinates of point XYZ and save them in stempoints\_no\_dup.txt.

1. Find\_path.m: find the shortest path between two points.

Recursively find the location of adjacent points, and record the shortest path to store a point and the previous point. Then judge whether each point has found the shortest path, and compare with each other to get the shortest path. If the current point cannot find the path to the lowest point, the shortest path from the set of found points to the set of not reached points is directly assigned. This is the key step. In the demo, we only calculate the Data term for improving our speed.

Input: load in stempoints\_no\_dup.txt or stempoints.txt.

Output: store all the path information and save them in path.txt.

1. Print\_path.m: print out all current paths.

Input: importing final\_data.txt and final\_path.txt.

Output: draw results according to your input data.

1. Optimized\_data.m: Optimize current path points based on the current path and modify the data. Optimize the paths one by one and check them one by one.

Input: importing stempoints\_no\_dup.txt and path.txt

Output: store the optimized data in optimal\_data.txt.

1. Insert\_points.m: based on the current path and the optimized points, interpolate and update the path.

Input: load in optimal\_data.txt and path.txt

Output: interpolated data and path are stored in insert\_data.txt and optimal\_path.txt

1. Remove\_path.m: delete unnecessary points based on the optimized path (mainly based on the path length, and the judgment method is to calculate the distance from the endpoint to the fork point)

Input: load in insert\_data.txt and optimal\_path.txt

Output: remove short paths and stored data in final\_data.txt

1. Connection\_points.m: analyze the path of the interpolated data again to generate skeleton.

Input: load in final\_data.txt

Output: final\_path.txt

1. Smooth\_path.m: smooth the path, the degree of smoothness is 0.1, the number of smoothness is 10, and the smoothing formula is L=1/2\*(nextpoint-currentpoint)+1/2\*(prepoint-currentpoint)

Input: load in final\_data.txt and final\_path.txt

Output: the data after smoothing is stored in smooth\_data.txt

Created by Xin Li

Email: lixin99@njfu.edu.cn

2021.11.22