**Task 1: Principal Component Analysis (PCA) Normal Estimation**

There are two parts in this task: (1) implement PCA function; (2) implement main function to compute the principal directions of the point cloud.

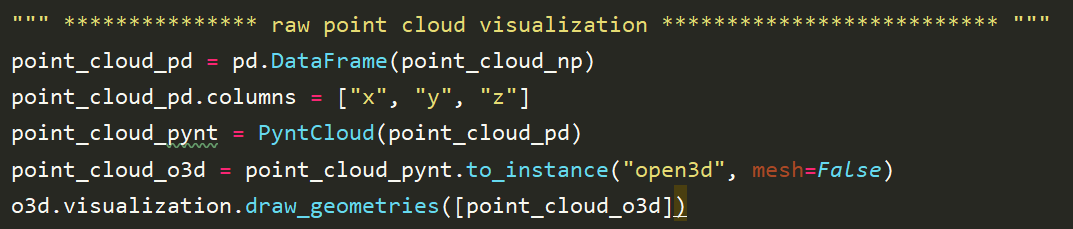
**Description of code snippet:**

1. define point cloud data path.
2. load point cloud data.
3. get coordinates from input data.

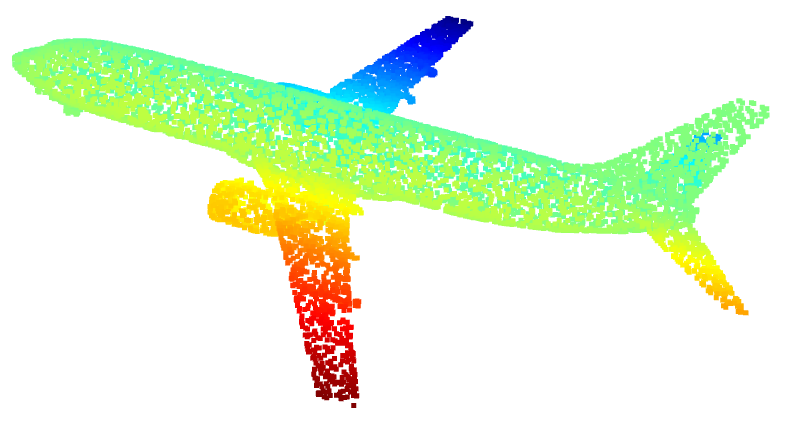
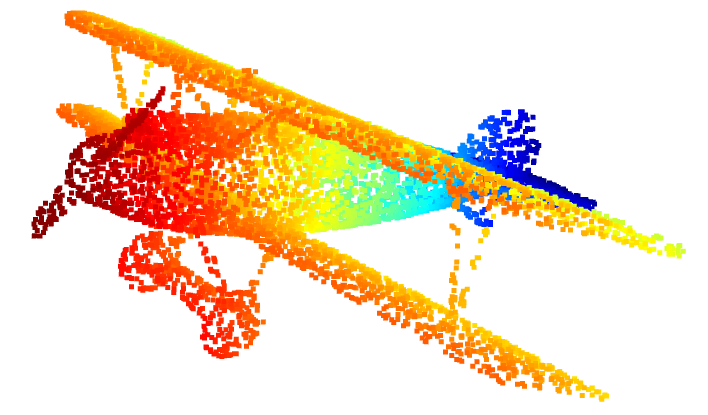
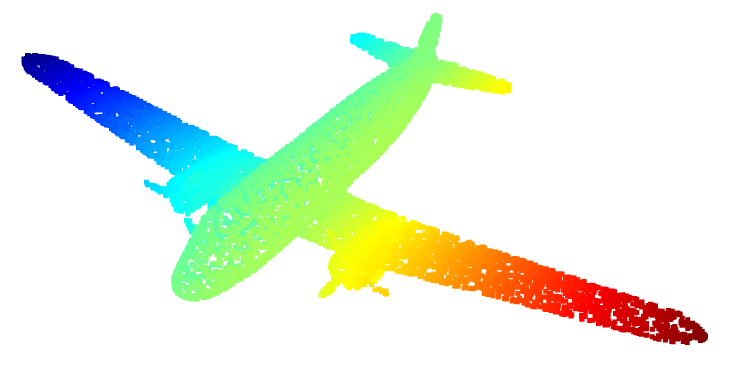


**Description of code snippet:**

1. convert Numpy format data to Pandas DataFrame data.
2. Convert DataFrame to point cloud format.
3. Instantiate point cloud data and visualize the raw point cloud data.

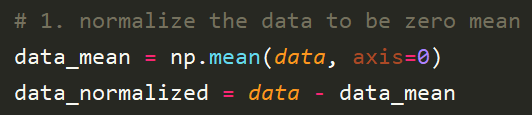
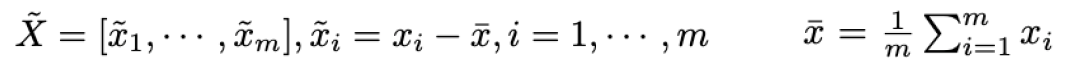


**Raw point cloud visualization:**

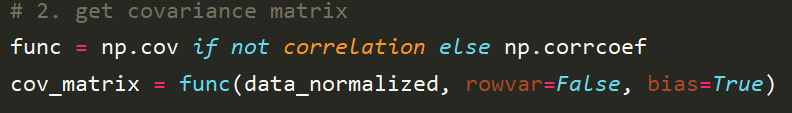
 

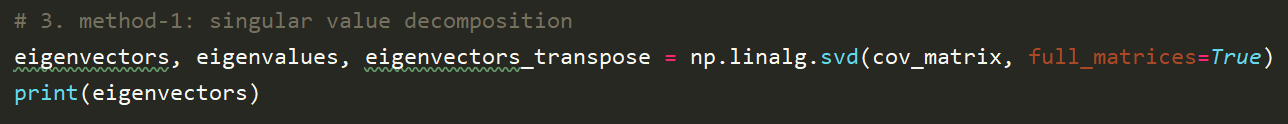
**Description of code snippet (PCA implementation):**

1. Normalization:

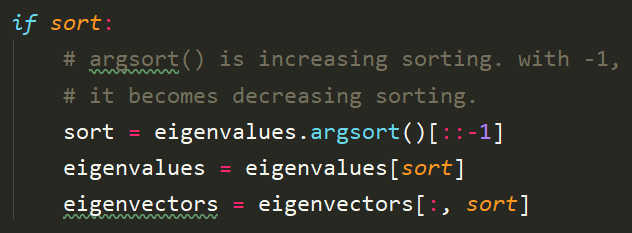
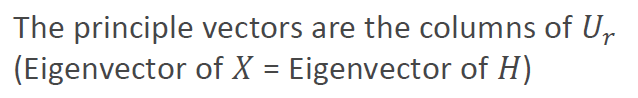


1. Compute covariance matrix and eigenvalue and eigenvectors using SVD

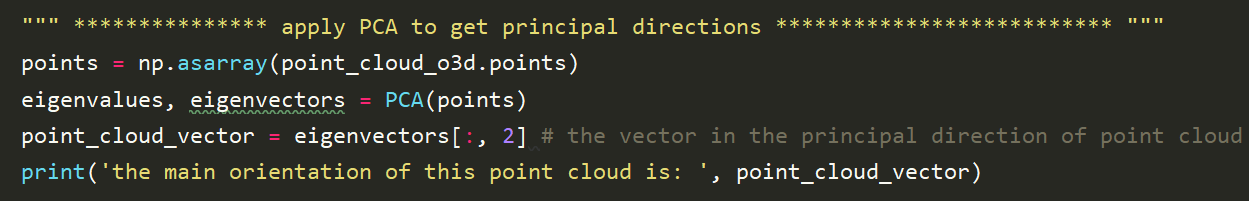




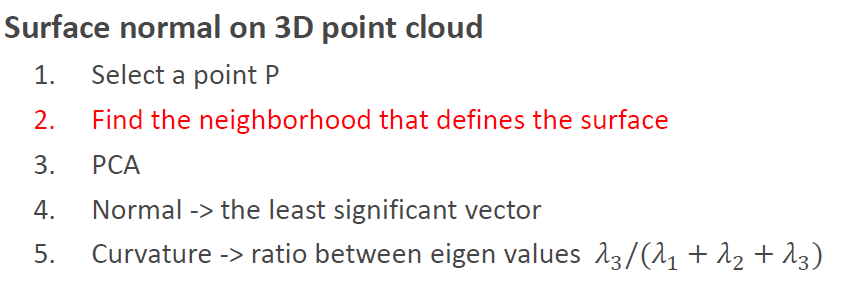
1. Decreasingly sort the eigenvalues and eigenvectors

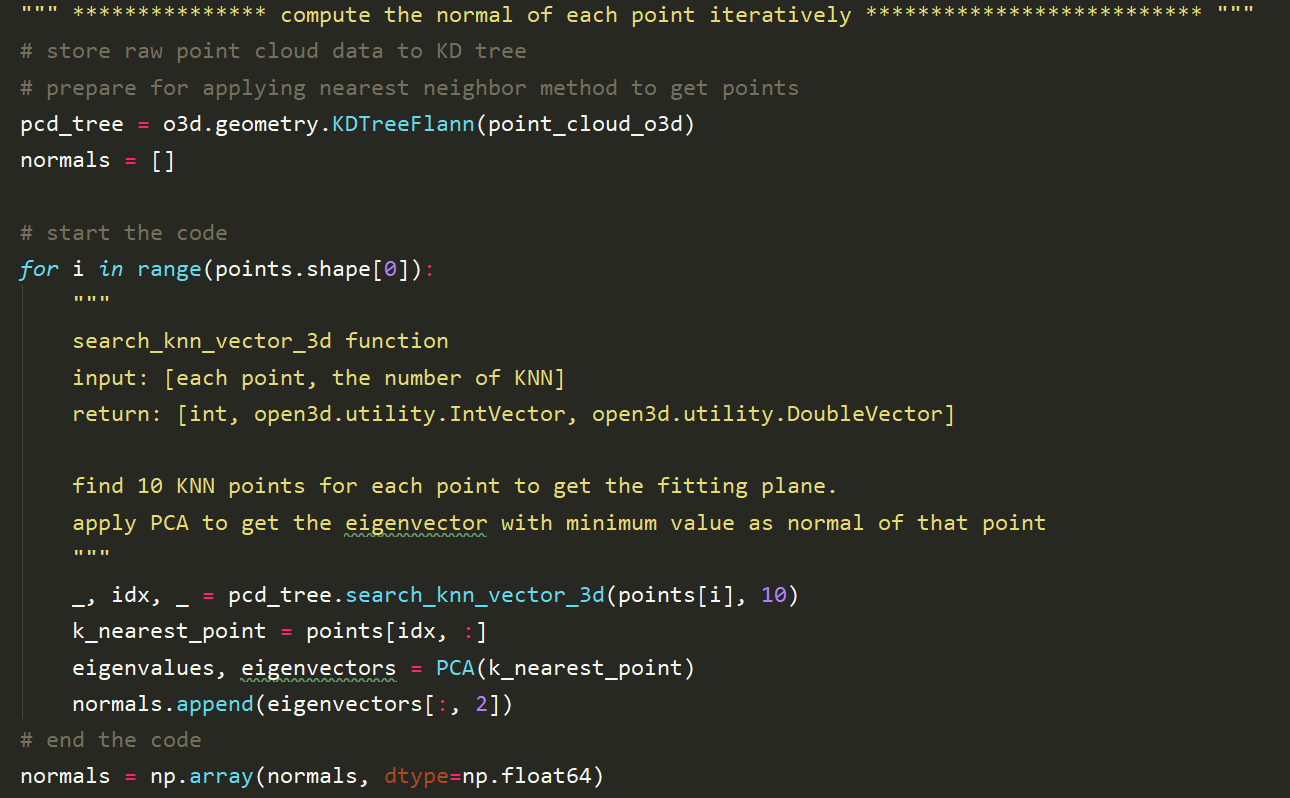


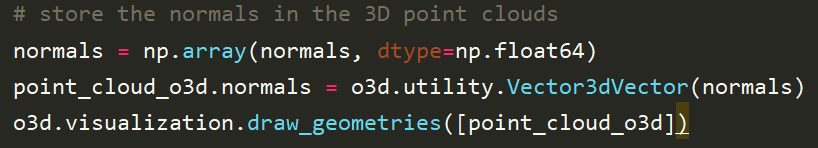
1. Apply PCA to get principal direction of point cloud



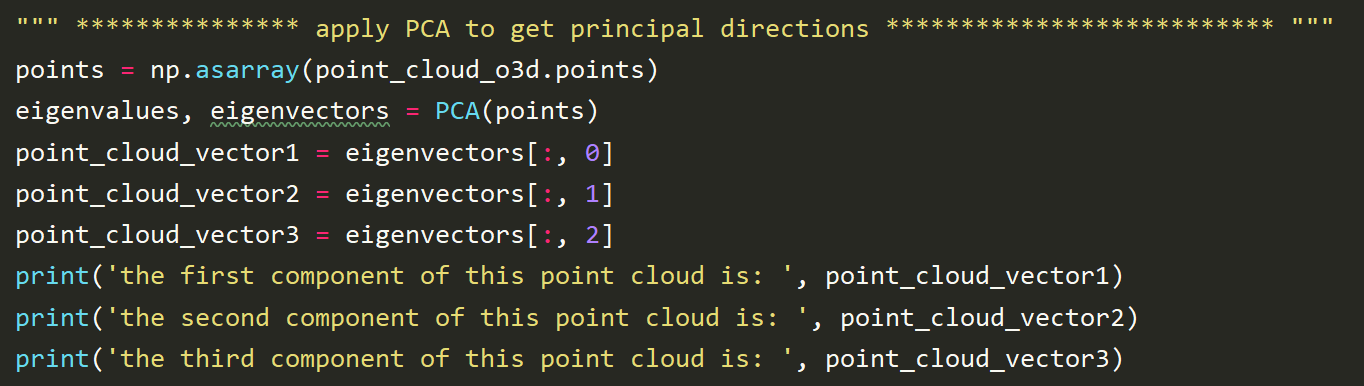
1. Compute surface normal using KNN

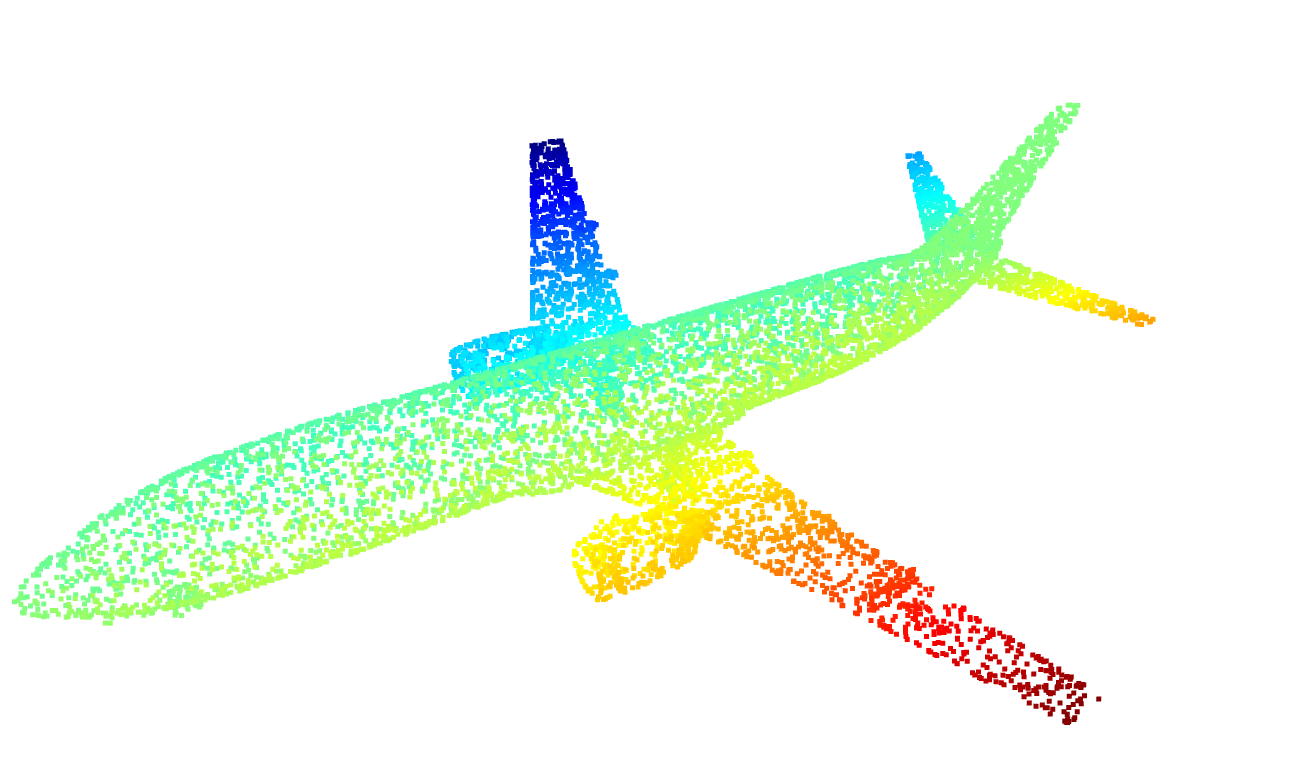
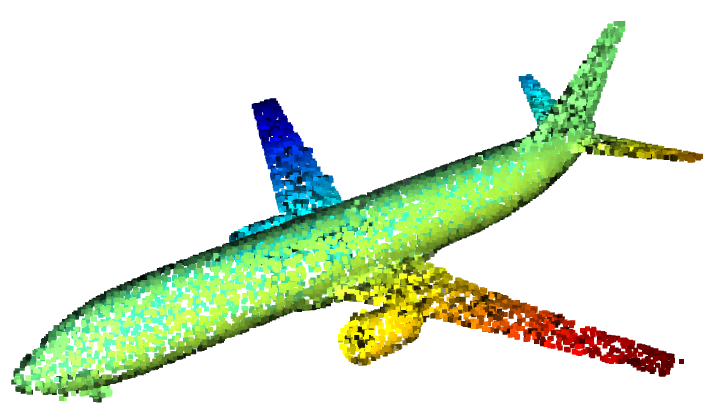
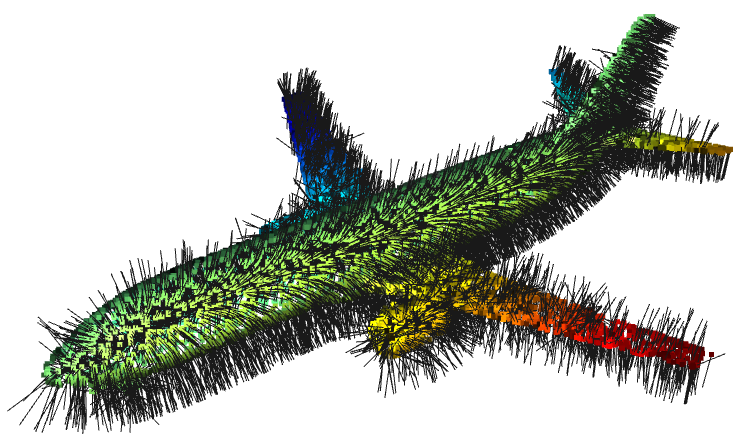




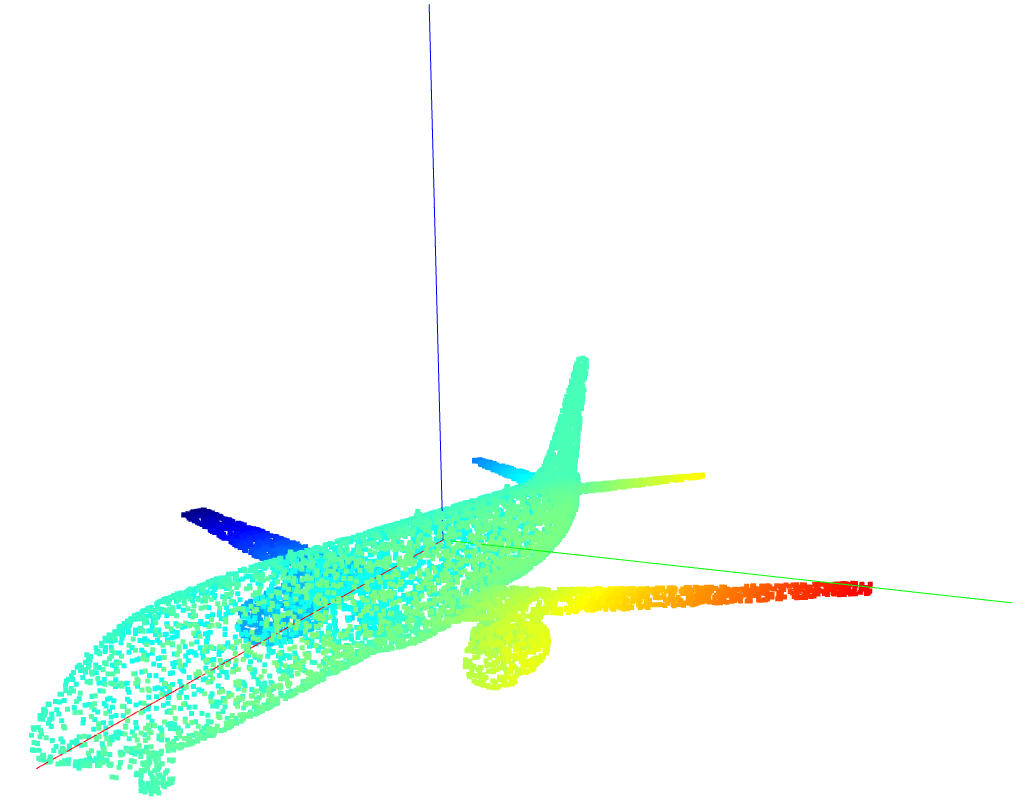


1. Results visualization

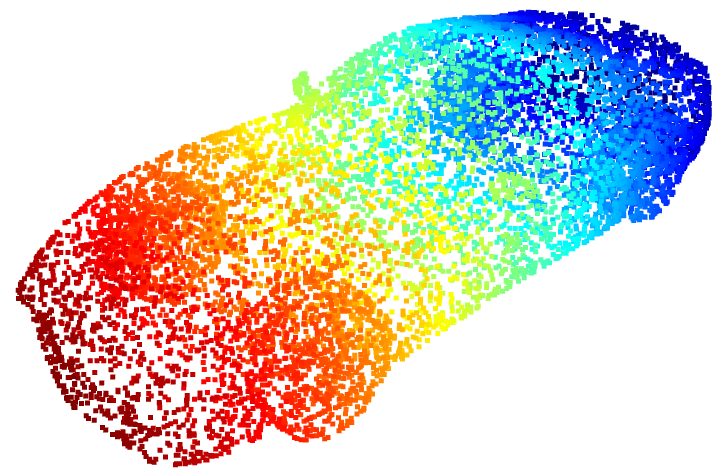
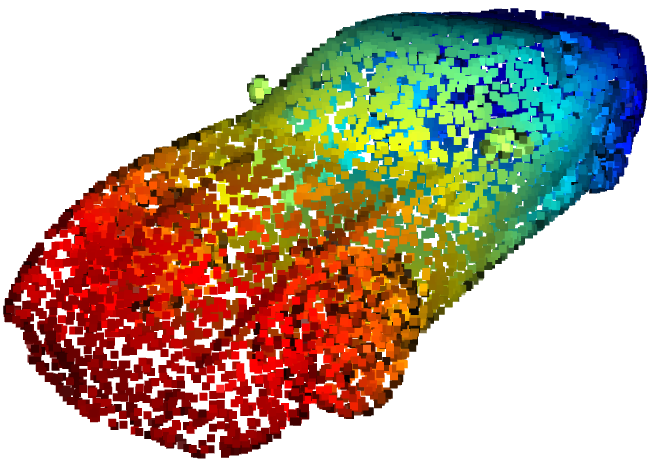
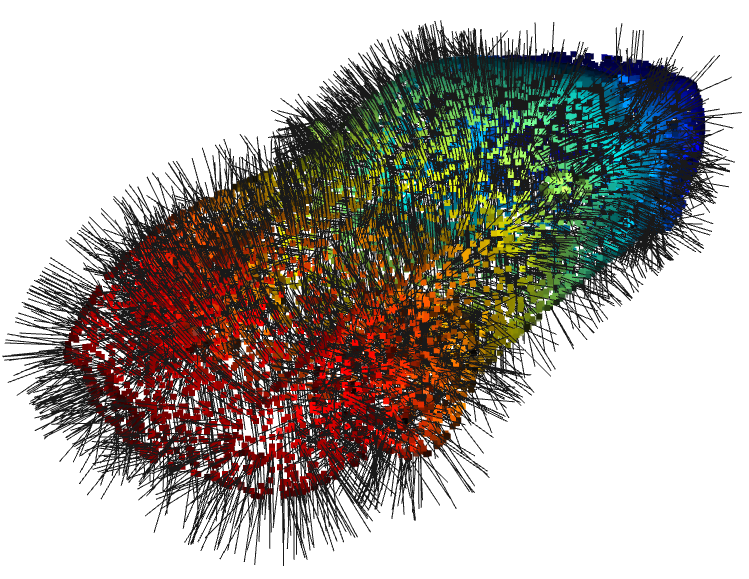


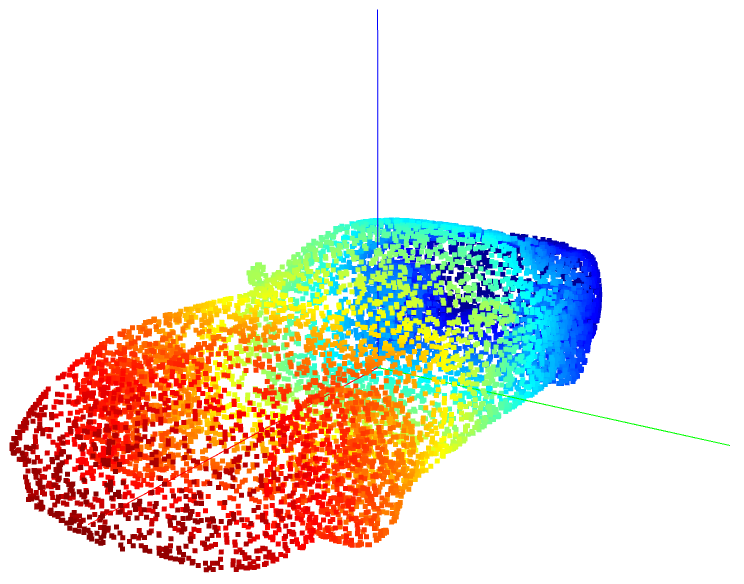
raw point cloud point cloud with noramlss point cloud with displaying normals



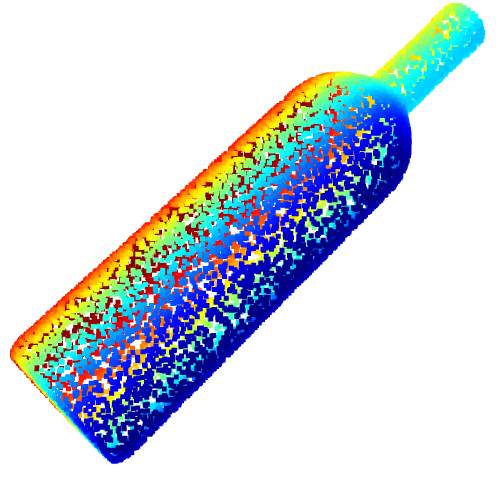
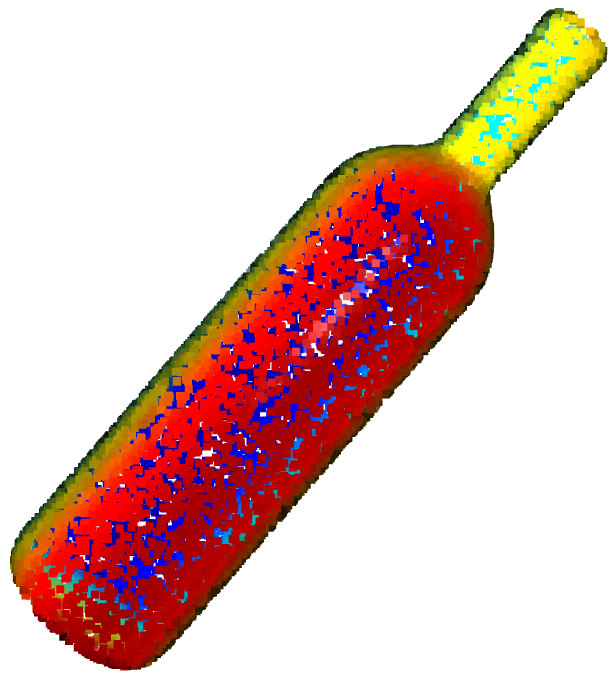
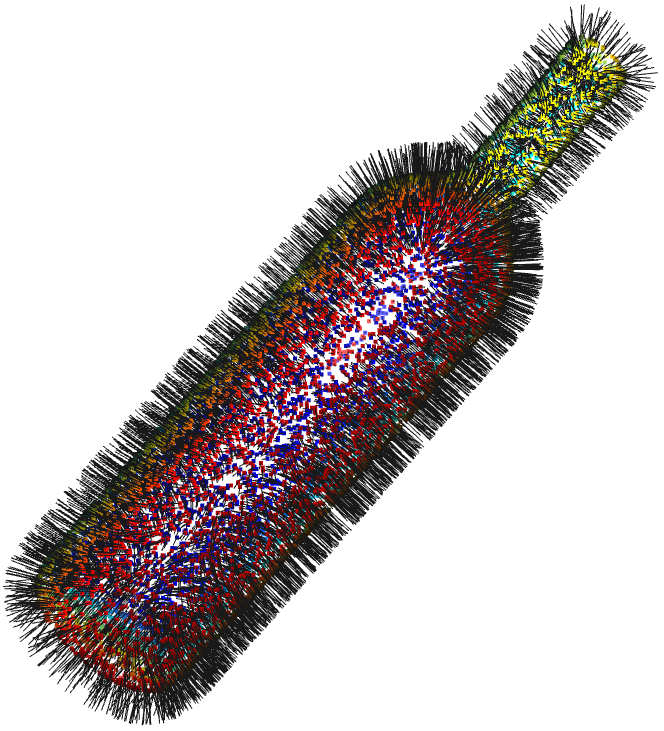
Point cloud with three principal components

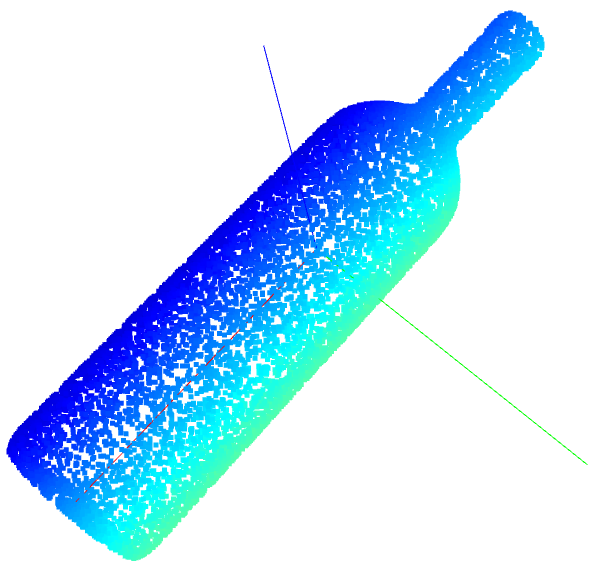
Raw point cloud point cloud with noramlss point cloud with displaying normals



Point cloud with three principal components

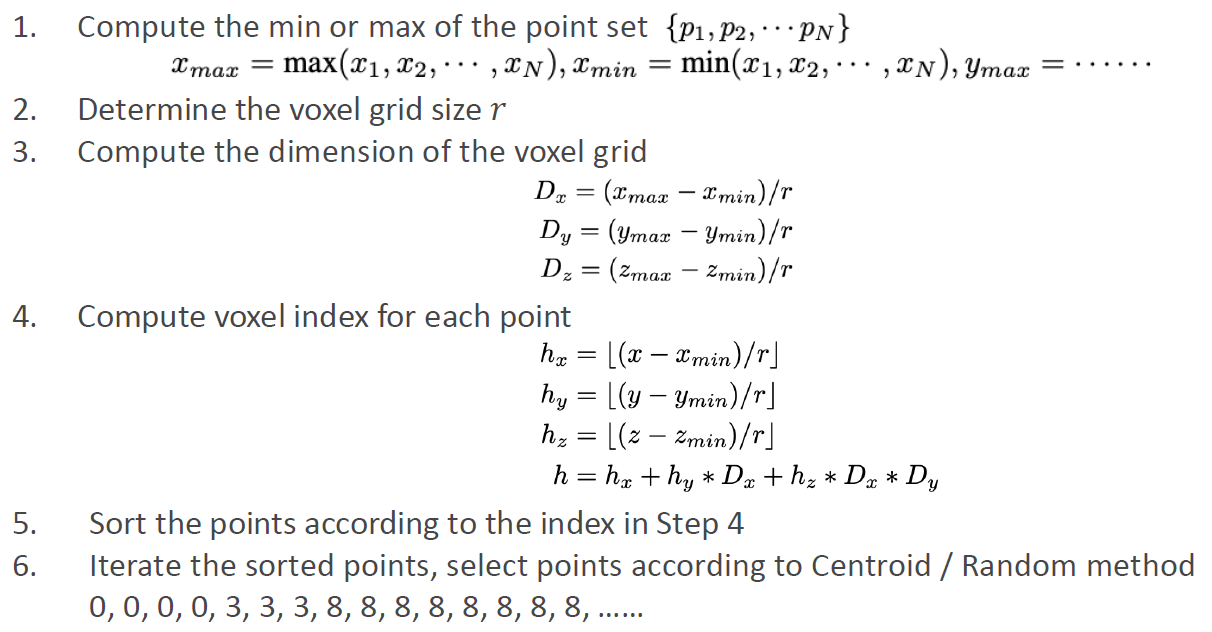
Raw point cloud point cloud with noramlss point cloud with displaying normals



Point cloud with three principal components

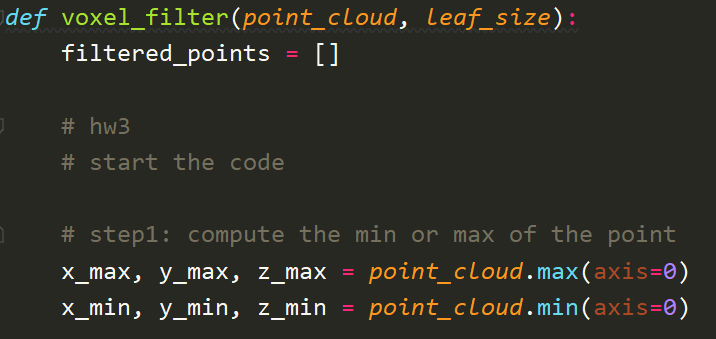
**Task 2: Voxel Filtering Down-sampling**

**Algorithm logic**:

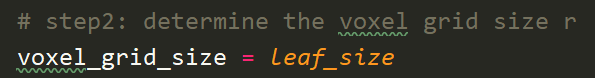


**Code Snippets:**

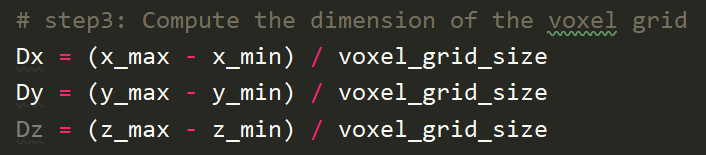
Step 1: Compute the minimum and maximum values of point cloud data



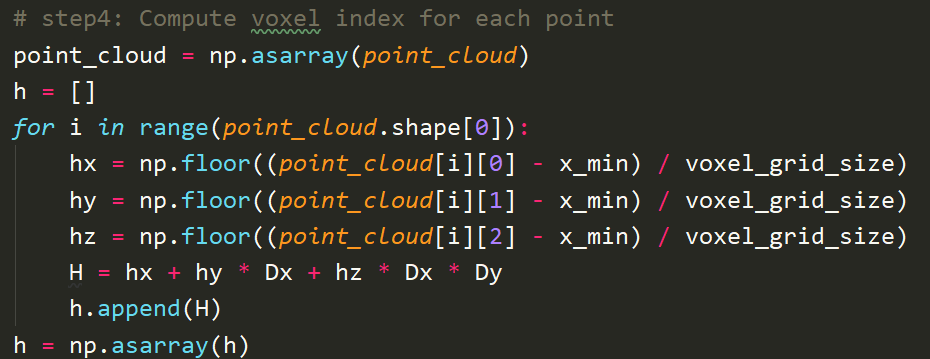
Step 2: assign voxel grid size from input



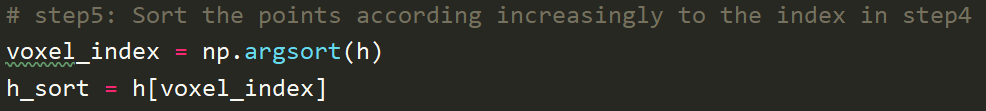
Step 3: compute the dimension of voxel grid



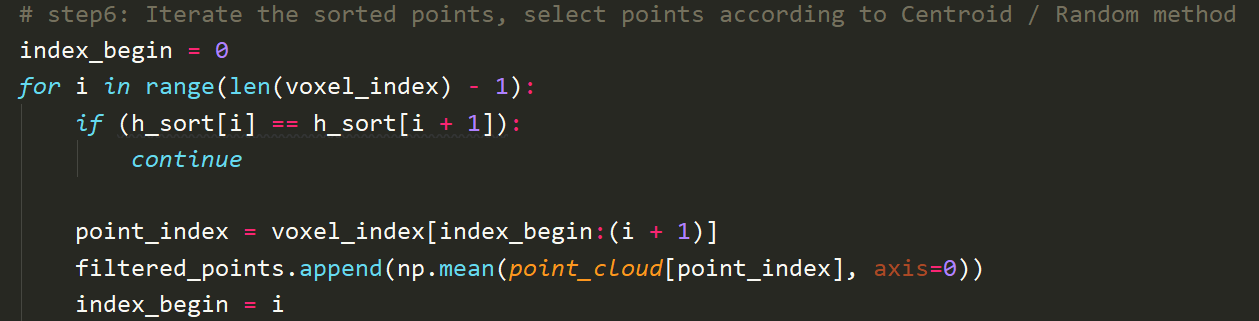
Step 4: compute voxel index for each point



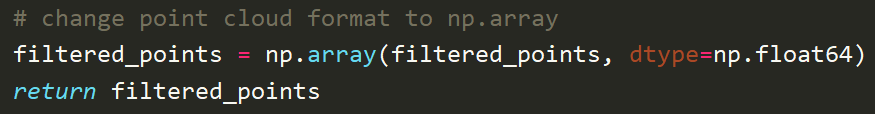
Step 5: Sor the points according to the index in Step 4



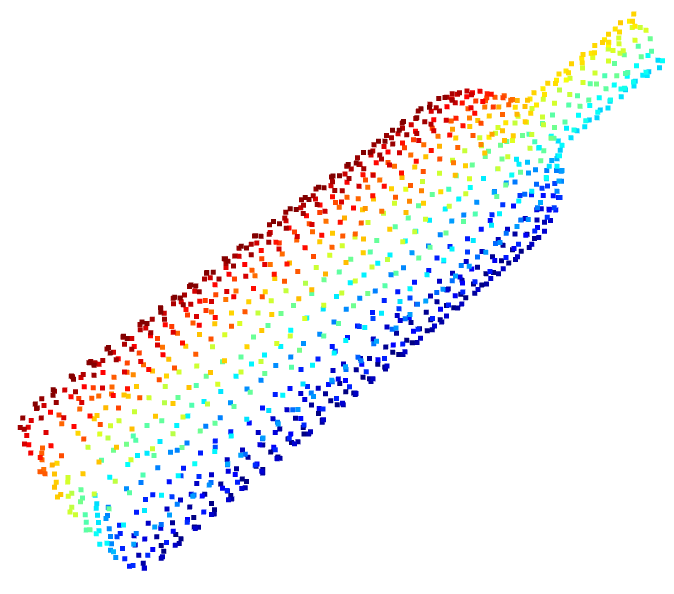
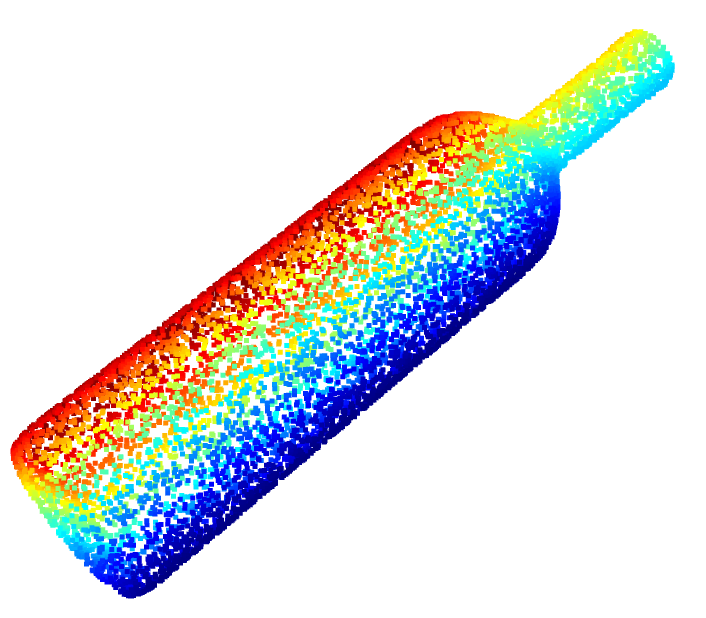
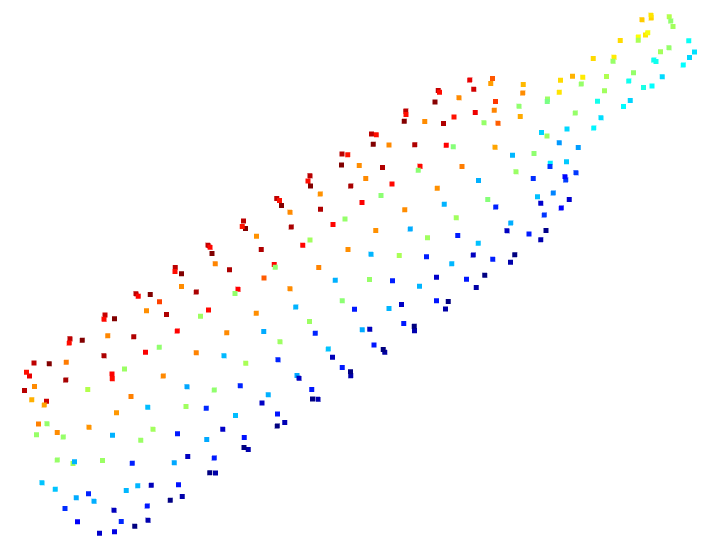
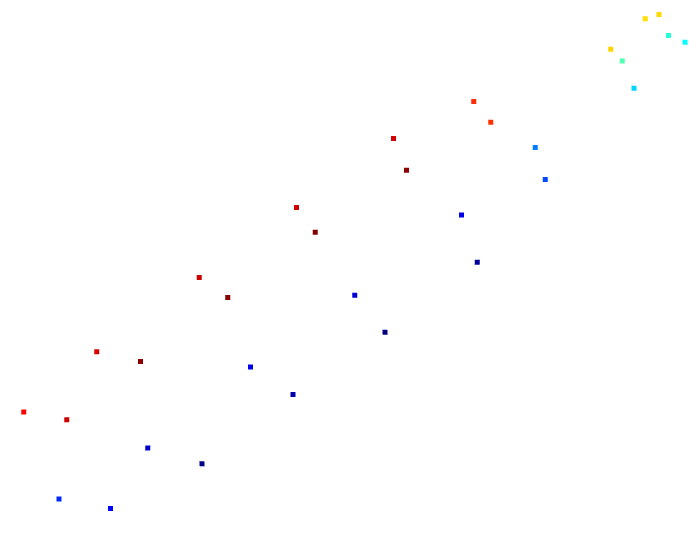
Step 6: Iterate the sorted points, select points according to Centroid / Random method



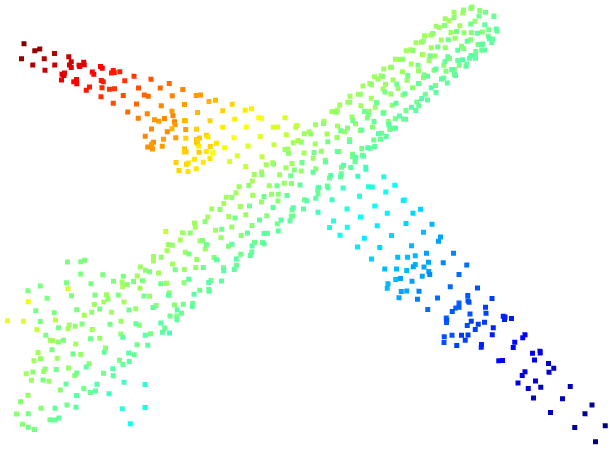
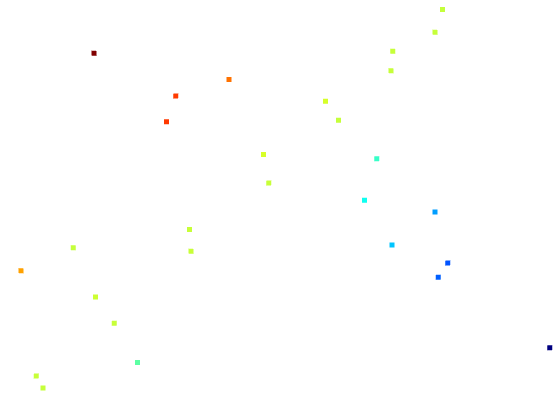
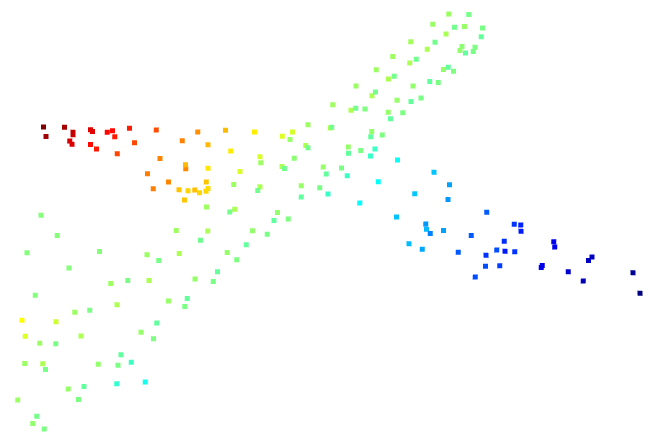
Return the result in float64 format which can avoid overflow.



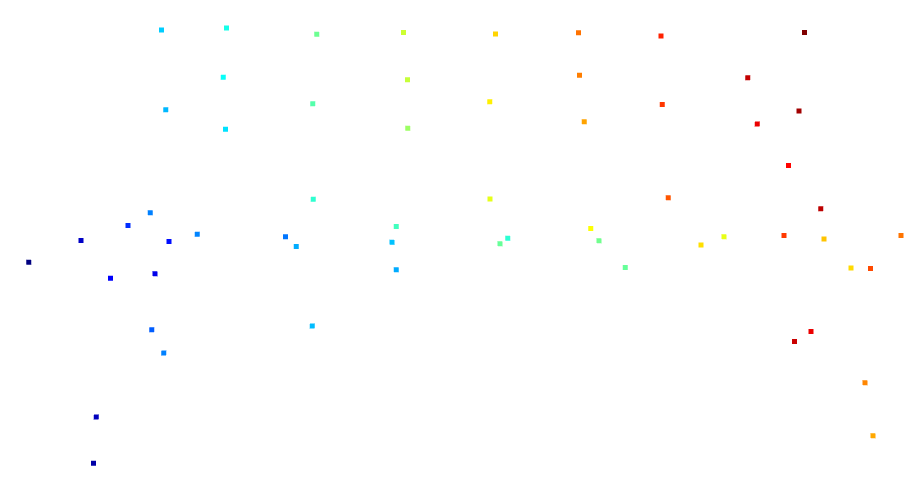
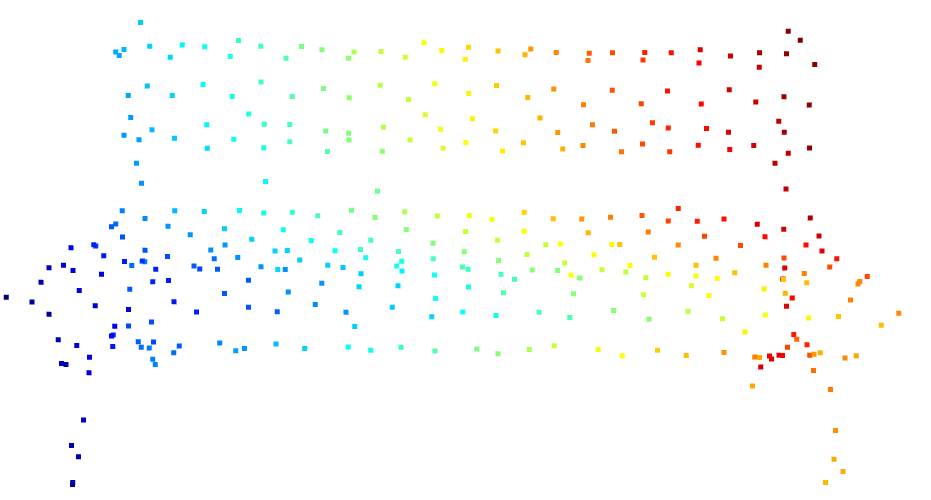
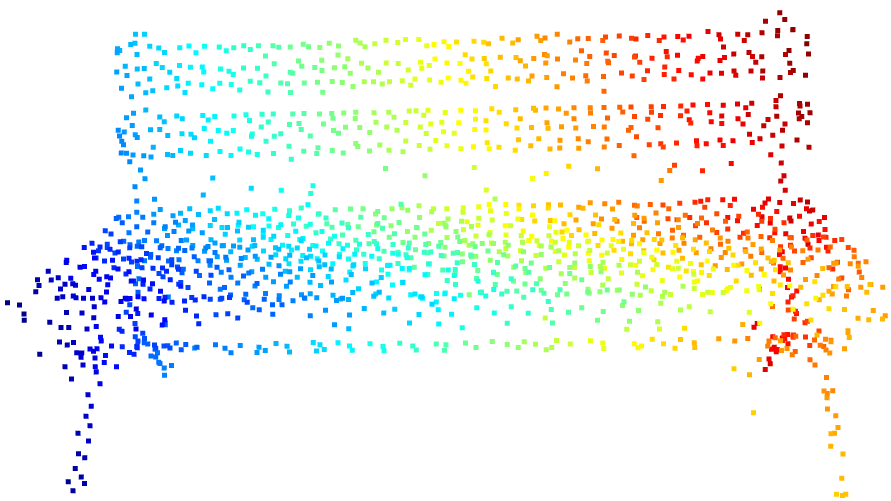
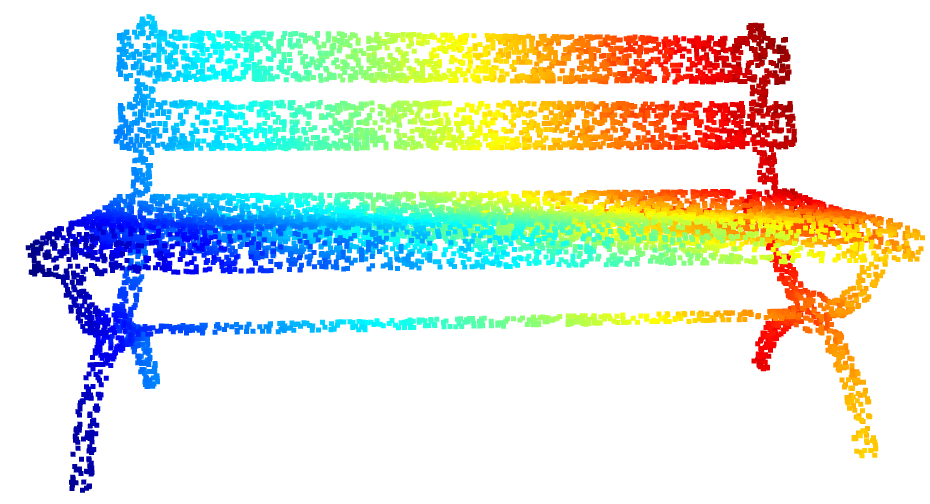
**Filtered Result Visualization:**

Raw point cloud Voxel size = 0.05 Voxel size = 0.1 Voxel size = 0.3

Raw point cloud Voxel size = 0.05 Voxel size = 0.1 Voxel size = 0.3



Raw point cloud Voxel size = 0.05 Voxel size = 0.1 Voxel size = 0.3