



深蓝学院  
shenlanxueyuan.com

## 三维点云处理第一次作业



主讲人 王永浩



# 第一题

## ●对点云数据进行PCA分析并进行投影可视化

```
def PCA(data, correlation=False, sort=True):  
    # 作业1  
    # 屏蔽开始  
  
    mean_vec = np.mean(data,axis=0)  
    normal_vec = data - mean_vec  
    H_vec = np.dot(normal_vec.T , normal_vec)  
  
    eigenvectors,eigenvalues,_ = np.linalg.svd(H_vec)  
  
    # 屏蔽结束  
  
    if sort:  
        sort = eigenvalues.argsort()[::-1]  
        eigenvalues = eigenvalues[sort]  
        eigenvectors = eigenvectors[:, sort]  
  
    return eigenvalues, eigenvectors
```

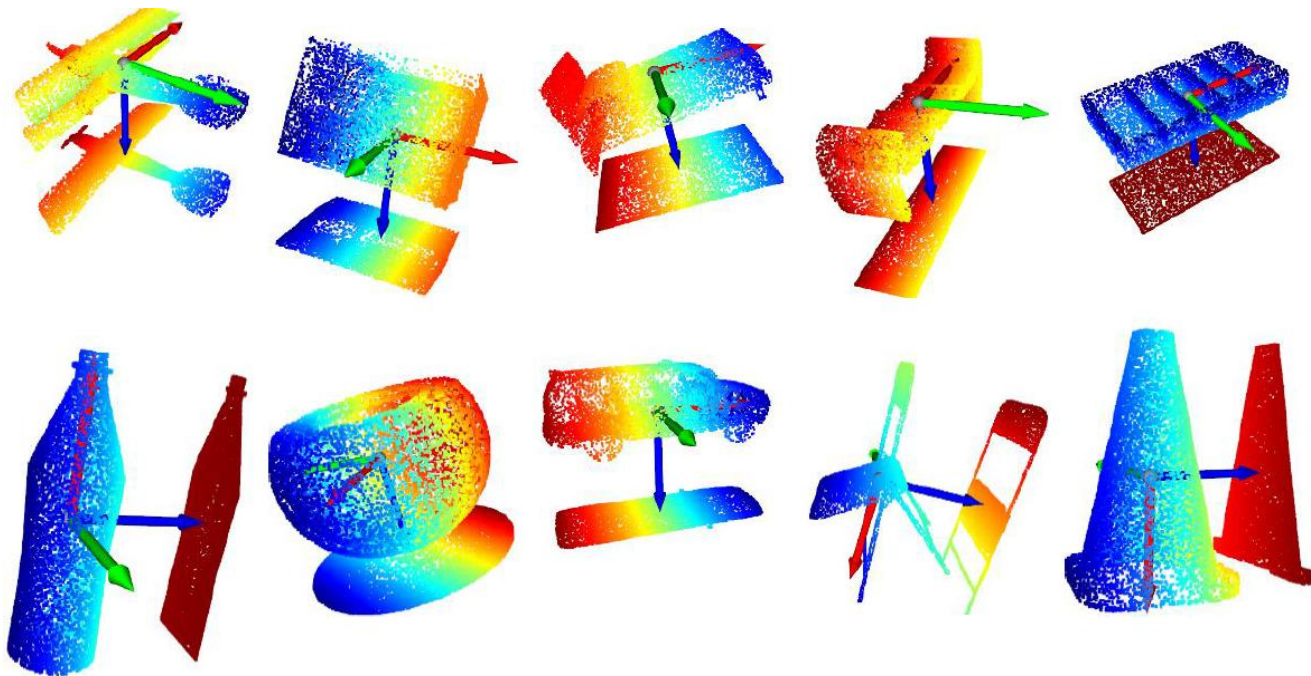
PCA分析

```
w,v = PCA(pointcloud)  
  
# PCA分析点云主方向  
pointcloud_vector = v[:,0]  
print('the main orientation of this pointcloud is: ', pointcloud_vector)  
  
axis = o3d.geometry.TriangleMesh.create_coordinate_frame().rotate(v,center=(0,0,0))  
  
pr_data2 = pointcloud - np.dot(pointcloud,v[:,2][:,np.newaxis])*v[:,2]  
  
pr_data2 = 1*v[:,2]+pr_data2  
  
principle_axis = np.concatenate((np.array([[0.,0.,0.]]),v.T))
```

将点云进行投影可视化

# 第一题

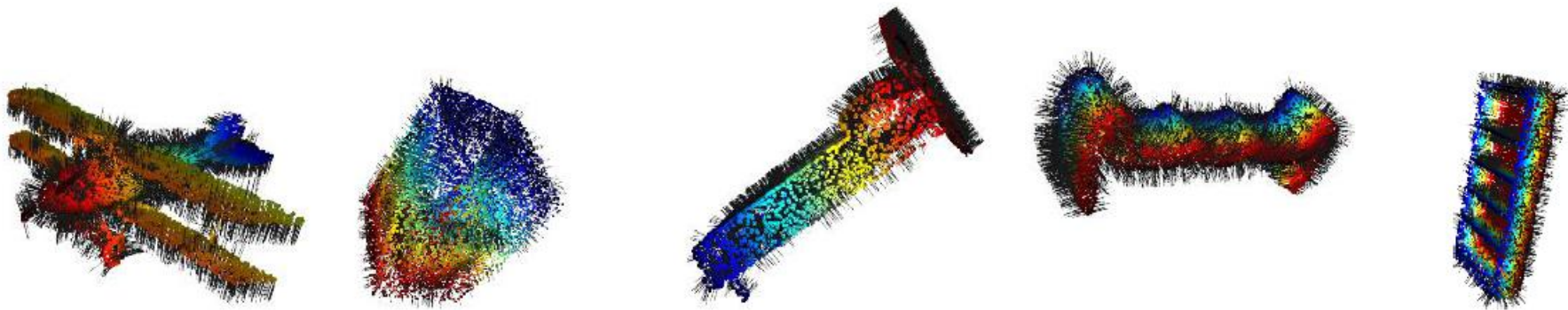
- 对点云数据进行PCA分析并进行投影可视化



# 第一题

## ● 利用PCA分析进行法向量估计

```
# 循环计算每个点的法向量
for index in range(pointcloud.shape[0]):
    [_,idx,_] = pcd_tree.search_knn_vector_3d(pc_view.points[index],k)
    neighbor_pc = np.asarray(pc_view.points)[idx]
    _,v = PCA(neighbor_pc)
    normals.append(v[:,2])
normals = np.array(normals,dtype=np.float64)
```



## 第二题

### ●体素式滤波

1. Numpy数组支持逻辑数组直接访问,不用进行sort类似的操作
2. 注意使用向量化编程的思想（类似matlab），少用for循环，可以加快速度



```
def voxel_filter(point_cloud, leaf_size, if_mean=False):
    filtered_points = []
    # 作业3
    # 屏蔽开始
    x_min, x_max, y_min, y_max, z_min, z_max = np.min(point_cloud[:,0]), np.max(point_cloud[:,0]), \
        np.min(point_cloud[:,1]), np.max(point_cloud[:,1]), np.min(point_cloud[:,2]), np.max(point_cloud[:,2])

    Dx, Dy, Dz = (x_max-x_min)/leaf_size, (y_max-y_min)/leaf_size, (z_max-z_min)/leaf_size

    min_vec = np.array([x_min, y_min, z_min])
    index = np.floor((point_cloud.copy()-min_vec)/leaf_size)
    h_index = index[:,0]+index[:,1]*Dx+index[:,2]*Dx*Dy

    for index in np.unique(h_index):
        point_chosed = point_cloud[h_index==index]
        if if_mean:
            filtered_points.append(np.mean(point_chosed,axis=0))
        else:
            filtered_points.append(point_chosed[np.random.choice(a=point_chosed.shape[0])])
    # 屏蔽结束

    # 把点云格式改成array, 并对外返回
    filtered_points = np.array(filtered_points, dtype=np.float64)
    return filtered_points
```



## 第二题

### ● 体素式滤波

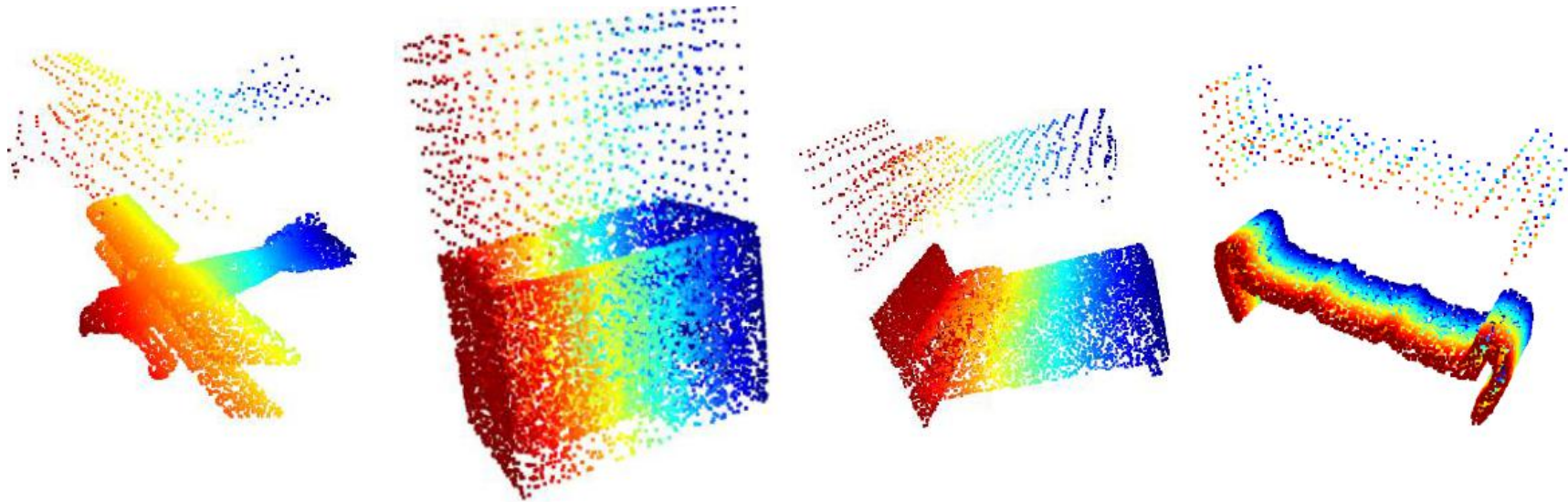


图 1.4 Centroid Method

## 第二题

### ● 体素式滤波

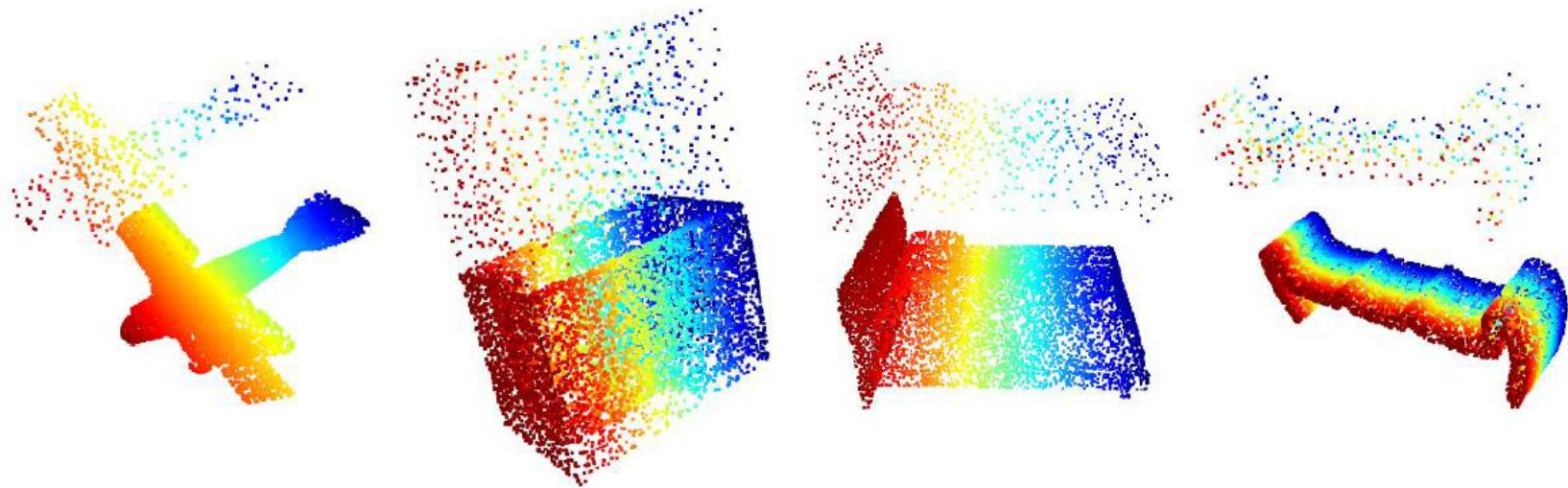


图 1.5 Random Method





感谢各位聆听 !  
Thanks for Listening

