import open3d as o3dimport numpy as npdef extract\_human\_points(cloud, eps=0.5, min\_points=50, max\_points=5000, visualize=False): """ Extract clusters of points corresponding to humans using DBSCAN. Args: cloud (o3d.geometry.PointCloud): Input point cloud. eps (float): Maximum distance between points to form a cluster. min\_points (int): Minimum points required to form a cluster. max\_points (int): Maximum points allowed in a cluster (to filter out large objects). visualize (bool): Whether to visualize the results. Returns: human\_clusters (list): List of point clouds corresponding to detected human clusters. """ # Compute clusters using DBSCAN labels = np.array(cloud.cluster\_dbscan(eps=eps, min\_points=min\_points)) # Extract clusters human\_clusters = [] for label in range(labels.max() + 1): cluster\_indices = np.where(labels == label)[0] cluster = cloud.select\_by\_index(cluster\_indices) # Filter clusters based on size if len(cluster.points) >= min\_points and len(cluster.points) <= max\_points: human\_clusters.append(cluster) if visualize: # Assign random colors to each human cluster for visualization clustered\_cloud = o3d.geometry.PointCloud() colors = np.random.rand(len(human\_clusters), 3) for i, cluster in enumerate(human\_clusters): cluster.paint\_uniform\_color(colors[i]) clustered\_cloud += cluster print(f"Detected {len(human\_clusters)} human clusters.") o3d.visualization.draw\_geometries([clustered\_cloud]) return human\_clustersif \_\_name\_\_ == "\_\_main\_\_": # Load the input point cloud (replace "example.pcd" with your point cloud file) cloud = o3d.io.read\_point\_cloud("example.pcd") print(f"Loaded point cloud with {len(cloud.points)} points.") # Extract human points human\_clusters = extract\_human\_points(cloud, eps=0.5, min\_points=50, max\_points=5000, visualize=True) # Save each detected human cluster as a separate PCD file for i, cluster in enumerate(human\_clusters): filename = f"human\_cluster\_{i + 1}.pcd" o3d.io.write\_point\_cloud(filename, cluster) print(f"Saved: {filename}")