Week 6:
Oriented Bounding Boxes
& moving to PCL 1.8

Bounding Boxes

- Not too difficult to implement: provided out of the box in recent versions of PCL

- based on the pcl::MomentOfInertiaEstimation functionality (in the features sub-library)

- Picks up where segmentation leaves off

- Represented in our codebase by a new BoundingBox data structure

BoundingBox code snippet

```cpp
MomentOfInertiaEstimation<PointXYZ> feature_extractor;
feature_extractor.setInputCloud (this->pointCloud);
feature_extractor.compute();

std::vector<float> moment_of_inertia;
std::vector<float> eccentricity;
pcl::PointXYZ min_point_AABB,
              max_point_AABB,
              min_point_OBB,
              max_point_OBB,
              position_OBB;

Eigen::Matrix3f rotational_matrix_OBB;
float major_value, middle_value, minor_value;
Vector3f major_vector, middle_vector, minor_vector;
Vector3f mass_center;

feature_extractor.getMomentOfInertia (moment_of_inertia);
feature_extractor.getEccentricity (eccentricity);
feature_extractor.getAABB (min_point_AABB, max_point_AABB);
feature_extractor.getOBB (min_point_OBB, max_point_OBB, position_OBB, rotational_matrix_OBB);
feature_extractor.getEigenValues (major_value, middle_value, minor_value);
feature_extractor.getEigenVectors (major_vector, middle_vector, minor_vector);
feature_extractor.getMassCenter (mass_center);
```
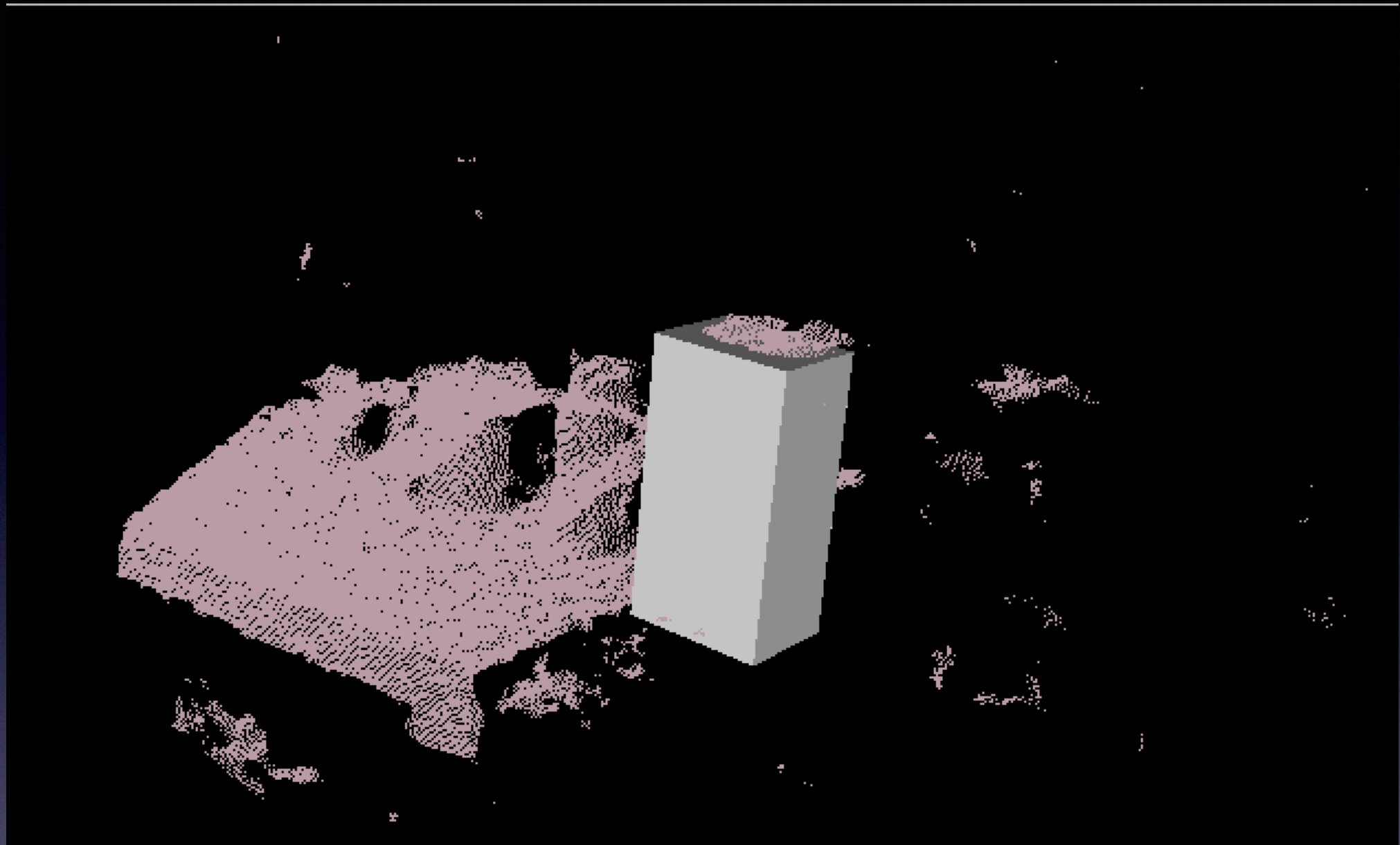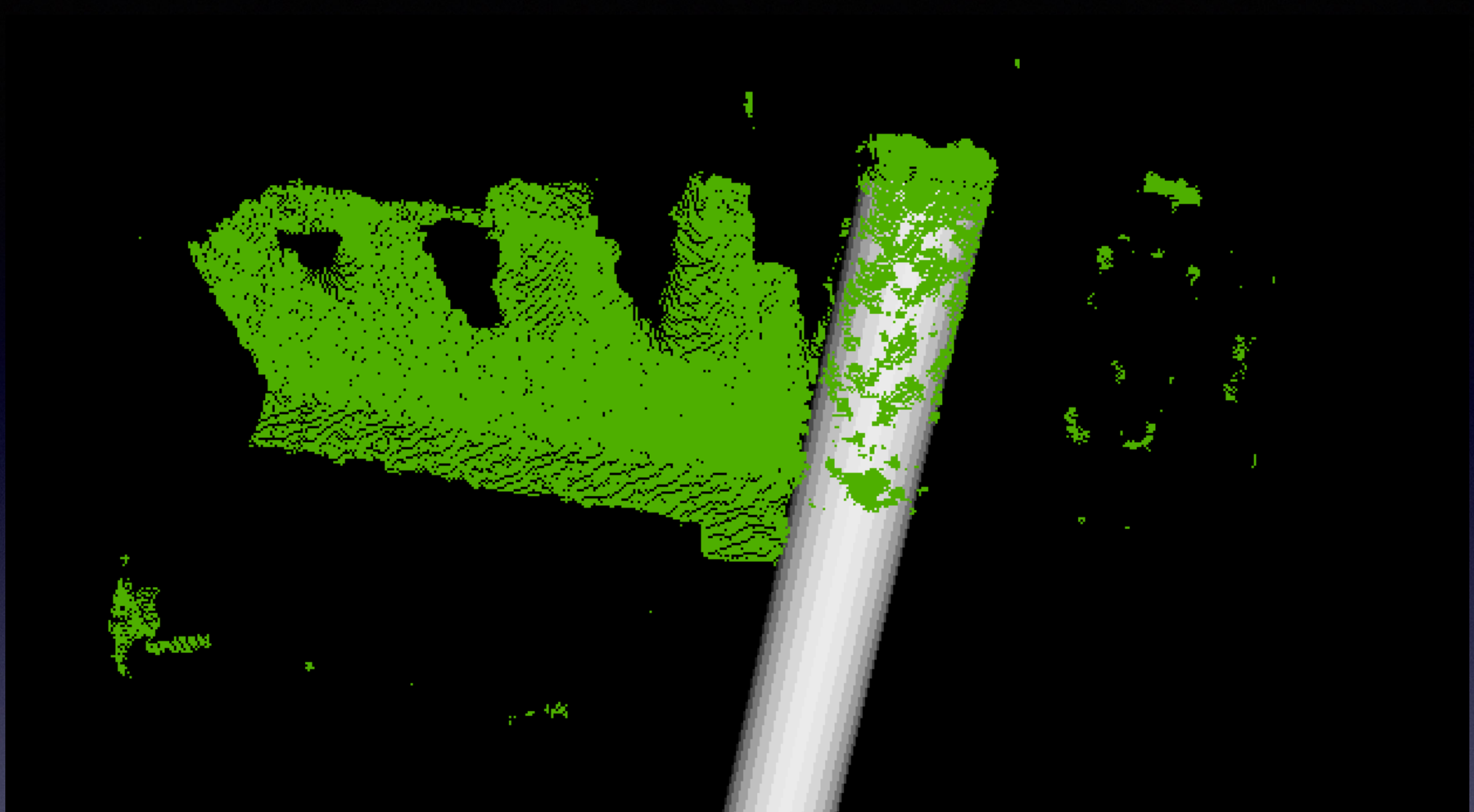
bounding box for a cylinder

Same image, showing cylinder co-efficients from segmentation, instead of bounding box of cloud

PCL 1.8 Release Candidate

PCL 1.8

- Motivation 1: the more features we get out of the box, the further we can get in creating new results for Intel

- Motivation 2: Some tutorials utilize features that are newer than what is in the current binaries

- Side-Benefit: we can debug/step into PCL source code, which will result in better understanding

- Drawback: building takes 30+ minutes on a current machine, and breaks a 4-5 year old machine