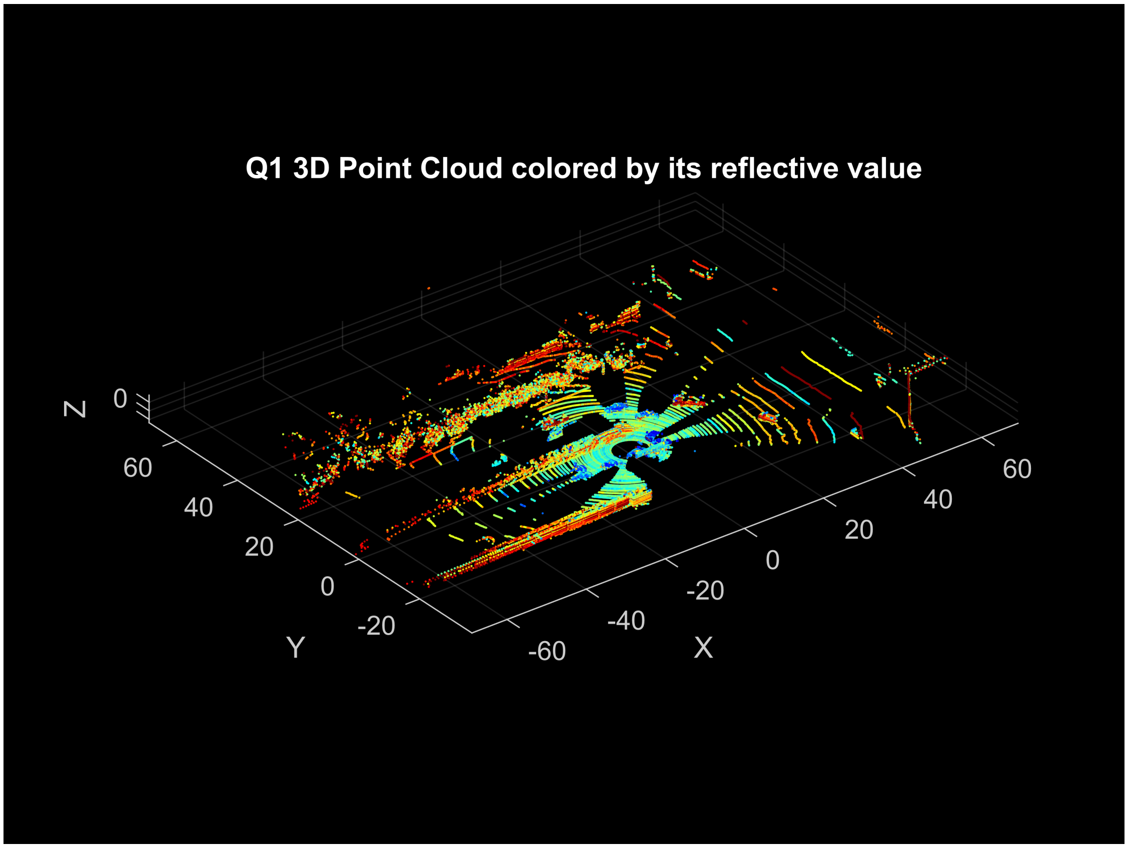
```
clear all
clc
```

Question (1):

Select a frame (or a few frames) of LiDAR data file, parse the file and visualize the 3D point cloud of this frame, colored by its reflectivity value.

```
%%Q1 Visualizing the 3D Point Cloud

% Reading the point cloud file
file_1 = fopen("C:\Users\siddh\OneDrive\Pictures\Screenshots\Documents\MATLAB\Perception\HW04_p
file_r = fread(file_1, 'float32');
n=1;
for i=1:4:length(file_r)
    x(n) = file_r(i);
    y(n) = file_r(i+1);
    z(n) = file_r(i+2);
    I(n) = file_r(i+3);
    n=n+1;

end

% Plotting the point cloud
figure(1)
Pt_cloud = pointCloud([x(:),y(:),z(:)],"Intensity",I(:));
pcshow(Pt_cloud);
xlabel('X');
ylabel('Y');
zlabel('Z');
title('Q1 3D Point Cloud colored by its reflective value')
```

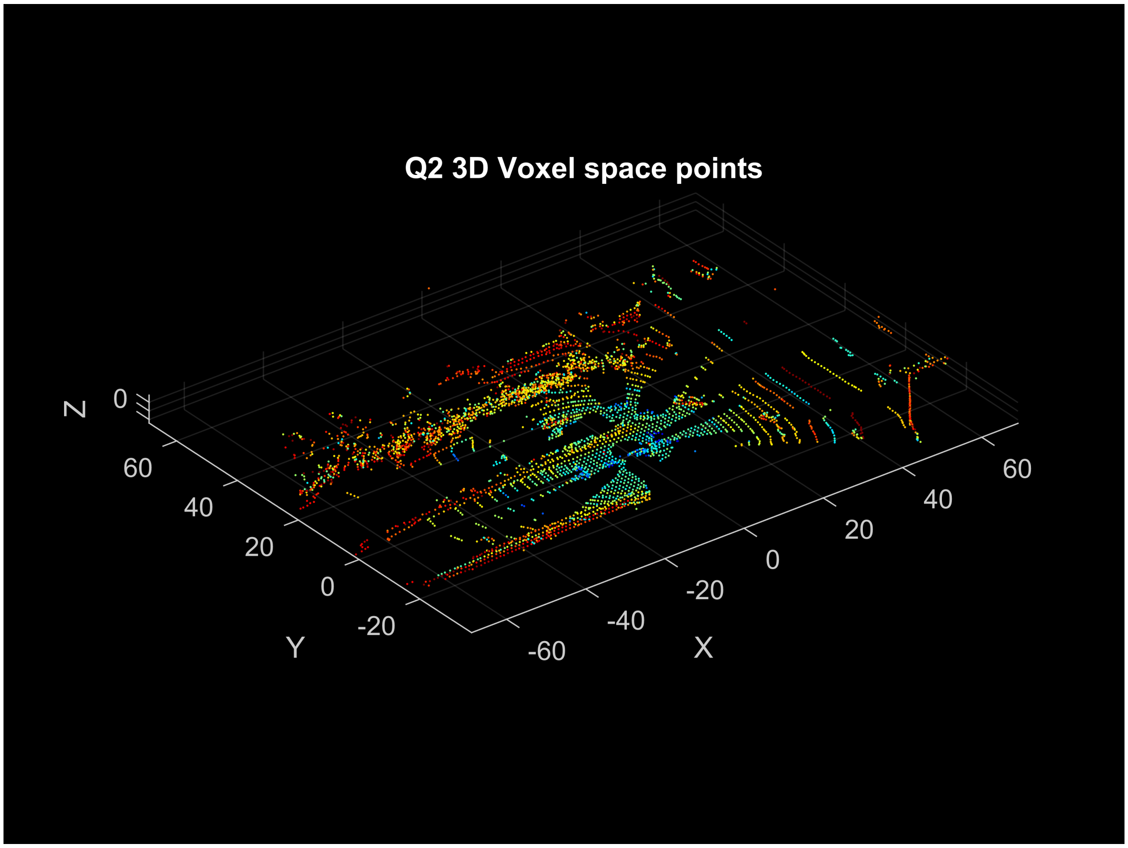



```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

Question (2)

Choose a 3-D resolution granularity, perform voxel filter (or box grid filter) to down-sample all the 3D point cloud points to the 3D voxel space points, and visualize the result points

```
%% Q2) Downsampling the 3D Point Cloud

% Applying box grid filter to the original point cloud
%Set the 3-D resolution to (1X1X1)
gridstep=1;
pt_cloud_voxel = pcdownsample(Pt_cloud,"gridAverage",gridstep);

% Plotting the points
figure(2)
pcshow(pt_cloud_voxel);
xlabel('X');
ylabel('Y');
zlabel('Z');
title('Q2 3D Voxel space points')
```




```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

Question (3)

- Apply RANSAC algorithm (or any others you prefer) to the 3D voxel space points to find a ground plane model. Print out your plane model parameter values result, visualize the plane with the points in the 3D.
- Analyze the computational time complexity of this algorithm.

Remove all the ground planes points in the 3D voxel space points, visualize all the off-ground points in the 3D

```
% 3.1 print model parameter
%Fit plane to 3-D point Data

%Set maximum point to plane distance (20cm) for plane fitting
maxDistance = 0.2;
%Set normal vecotor of the plane
referenceVector = [0,0,1];
%Set maximum angular ditance to 5 Degrees
maxAngularDistance = 5;

% 3.1 Applying MSAC algorithm to find a ground plane model.
% The MSAC algorithm is a variant of the RANdom SAmple Consensus (RANSAC) algorithm.
tic;
[model, in_ind, out_ind, error] = pcfitplane(pt_cloud_voxel,...
                               maxDistance, referenceVector, maxAngularDistance);
toc;
```

```
Elapsed time is 0.055356 seconds.
```

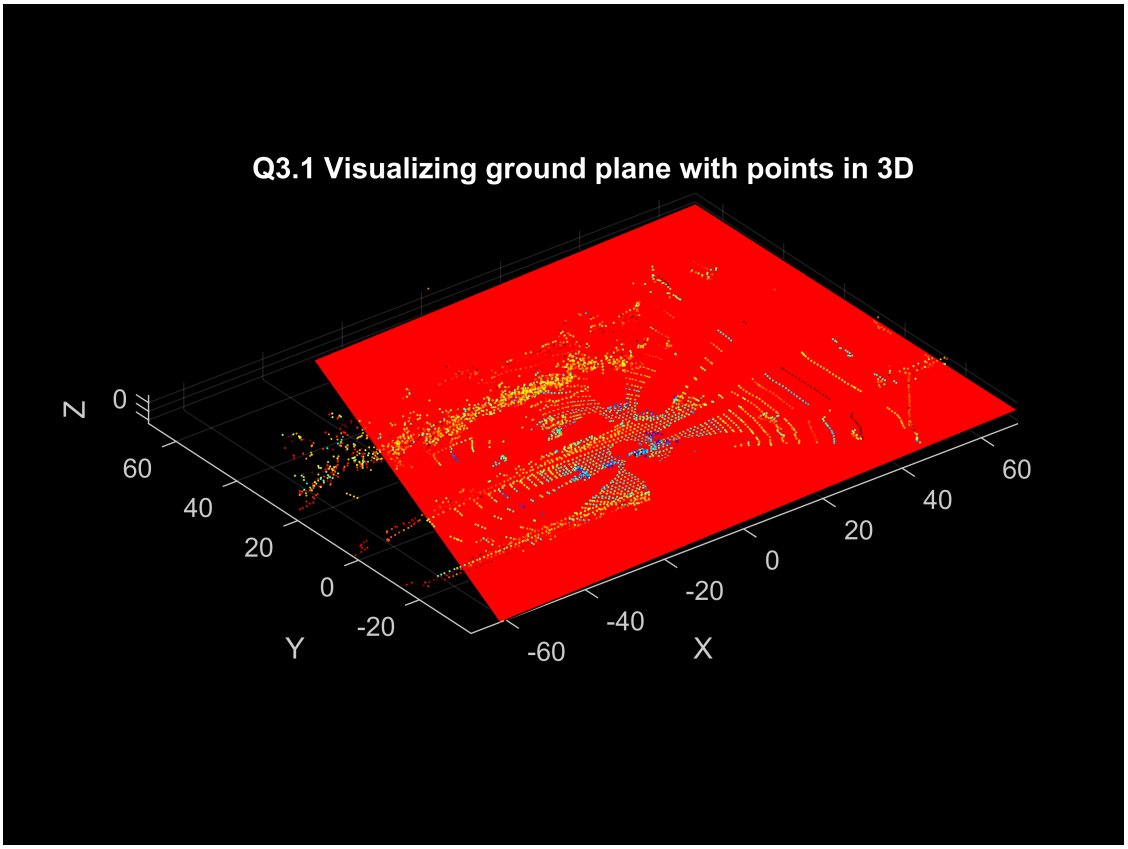
```
elapsedtime=toc;

% 3.1 Model Parameters
disp("Q3.1 The model parameters are:" )
```

```
Q3.1 The model parameters are:
```

```
disp(model.Parameters)
```

```
   -0.0242    0.0079    0.9997    1.5789
```

```
% 3.1 Plotting the ground plane model
figure(3)
pcshow(pt_cloud_voxel);
hold on
model.plot;
xlabel('X');
ylabel('Y');
zlabel('Z');
title('Q3.1 Visualizing ground plane with points in 3D')
hold off
```



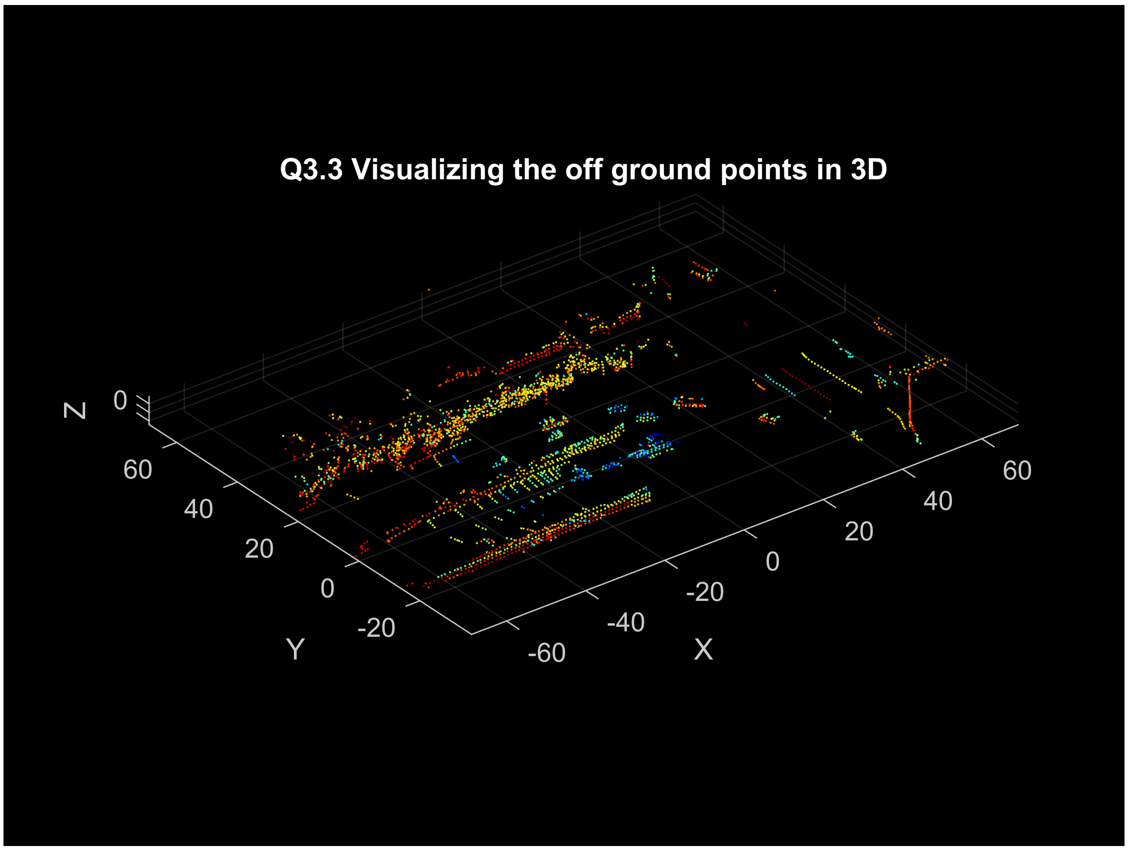

```
%3.2 Complexity of the Algorithm
disp("Q 3.2 Computational time complexity of Algorithm:" )
```

Q 3.2 Computational time complexity of Algorithm:

```
disp(elapsedtime)
```

```
    0.0603
```

```
% 3.3 Plotting the off ground points
figure(4)
groundoff_points = select(pt_cloud_voxel, out_ind);
pcshow(groundoff_points);
xlabel('X');
ylabel('Y');
zlabel('Z');
title('Q3.3 Visualizing the off ground points in 3D')
```



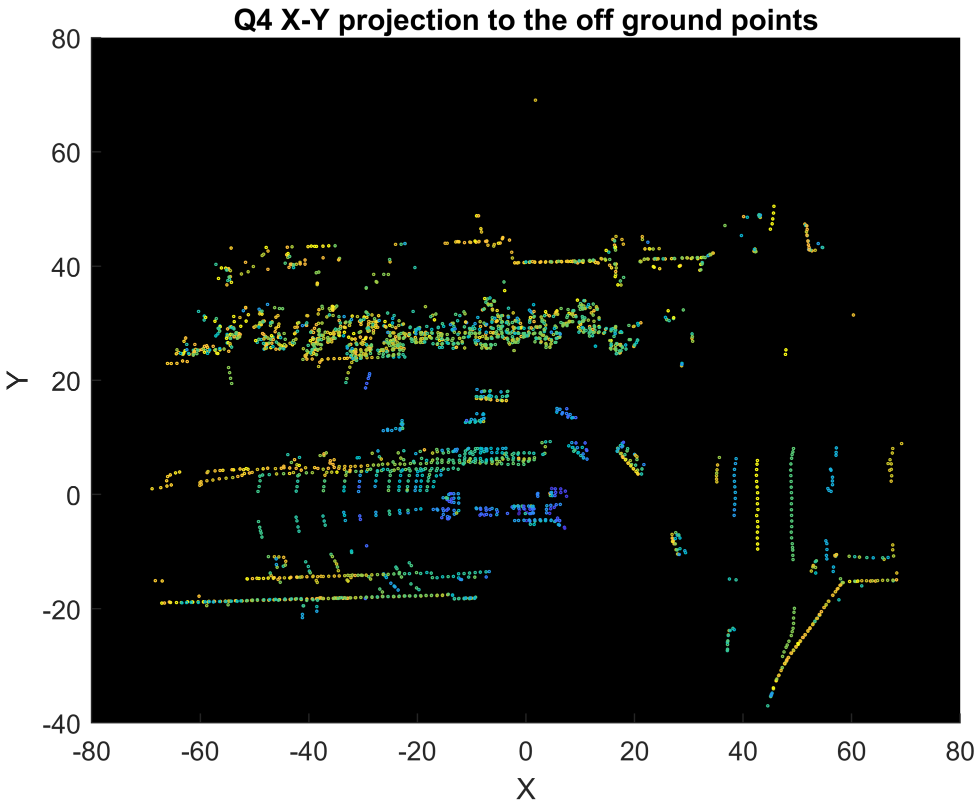

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

Question (4)

Perform a x-y projection to the off-ground points, and get a 2D matrix (you decide what is the element value), and visualize the 2D matrix as an image.

```
% Q4 X-Y projection of the off-ground points

off_ground = groundoff_points.Location;
Intensity = groundoff_points.Intensity;
```

```
figure(6)
% Convert Intensity values to color
color = Intensity(:);
scatter(off_ground(:,1), off_ground(:,2), 0.7, color);
set(gca, 'color', 'k');
xlabel('X');
ylabel('Y');
title('Q4 X-Y projection to the off ground points')
```



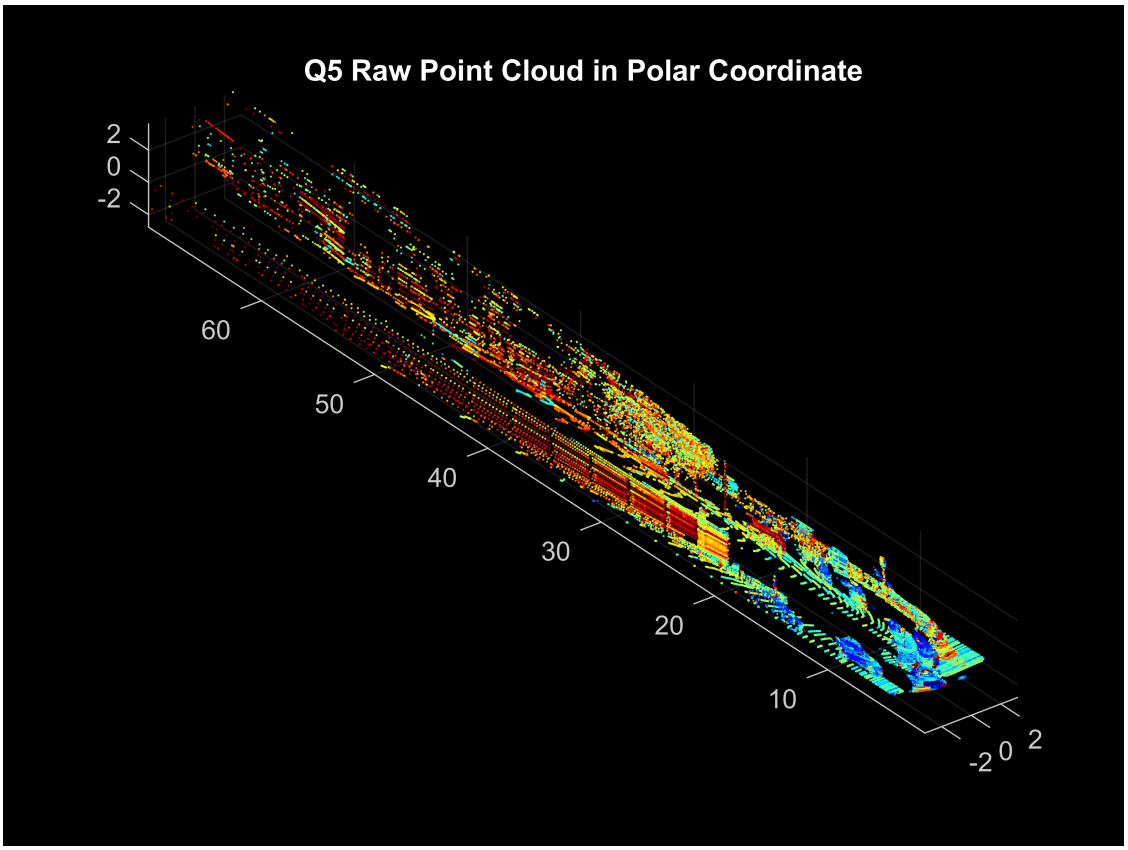

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

Question (5)

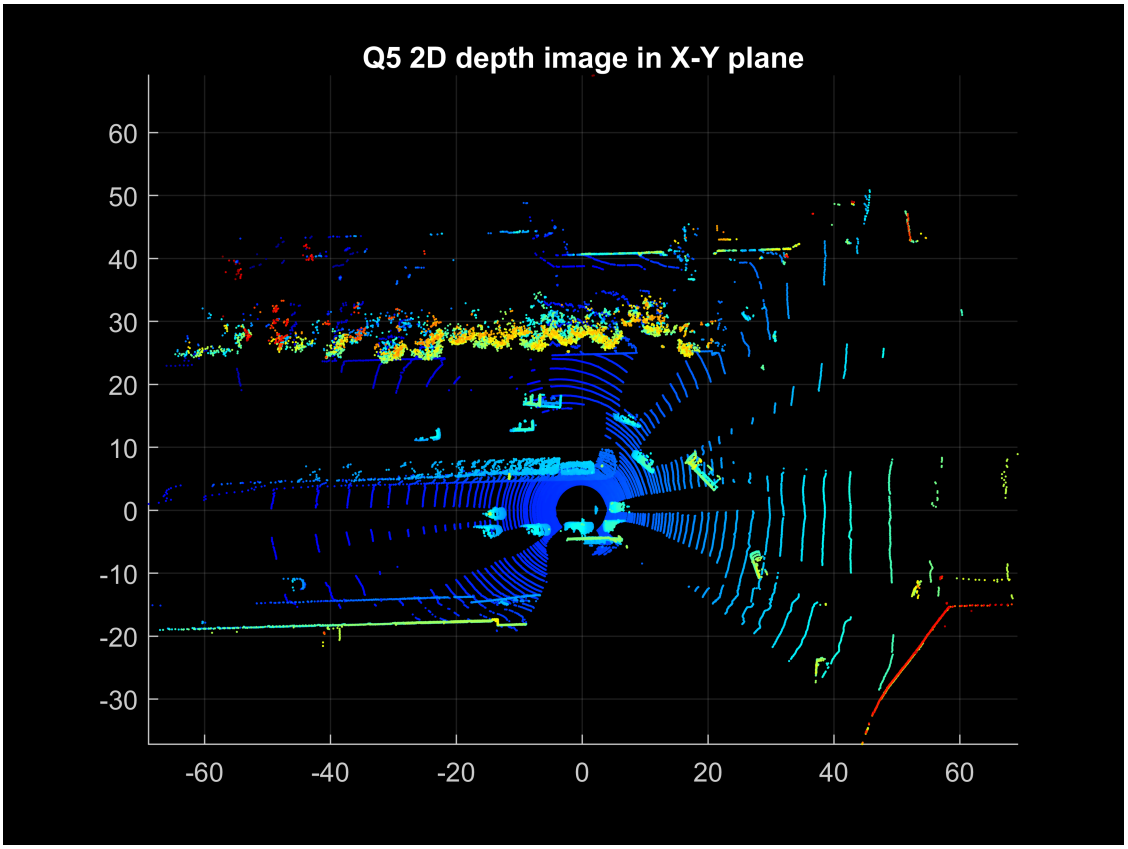
- Based on the raw point cloud data (Questions 1), which is in Cartesian Coordinate, represent and visualize all the point cloud in Polar Coordinate (with horizontal and vertical angels and distance to the original) (5 pts).
- Finally, generate the projected 2D depth image w.r.t horizontal and vertical angels, with intensity value using the distance. Visualize the 2D depth image (5 pts).

```
%% Q5 Raw Point Cloud date in Polar Coordinate

% Transform the pointcloud from cartesian to polar coordinate
[r1, r2, h] = cart2pol(x(:),y(:),z(:));

% Pointcloud for polar coordinate
polar_pt_cloud = pointCloud([r1(:),r2(:),h(:)], ...
```

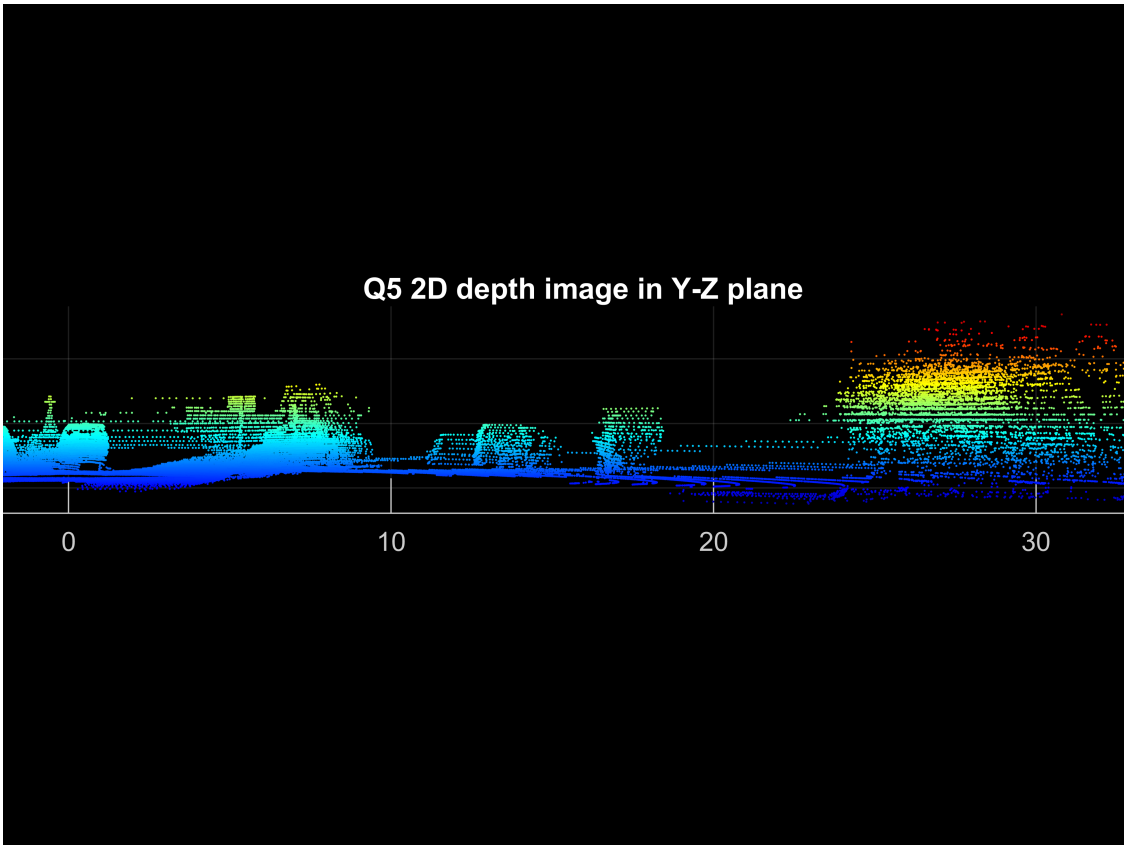
```
                    "Intensity",I(:));


% Visualizing point cloud in polar coordinate
figure(7)
pcshow(polar_pt_cloud);
title('Q5 Raw Point Cloud in Polar Coordinate')
```




```
% Visualizing the 2D depth image with intensity values as h
% X-Y Plane
figure(8);
point_cloud_2D = pointCloud([x(:),y(:),z(:)],'Intensity',z(:));
pcshow(point_cloud_2D);
view(0,90);
title('Q5 2D depth image in X-Y plane');
```




```
% Y-Z plane
figure(9);
point_cloud_2D = pointCloud([x(:),y(:),z(:)],'Intensity',z(:));
pcshow(point_cloud_2D);
view(90,0);
title('Q5 2D depth image in Y-Z plane');
zoom(5)
```


Q5 2D depth image in Y-Z plane
0
10
20
30