Assignment-2

LIDAR

Darshitaben Patel - pate2126

12th April 2024

ANSWER

Generate point density maps for the two point clouds with the following cell sizes: 0.5 m and 1 m.

For file: 20201002_VLP32C_ICSC_ds5.las

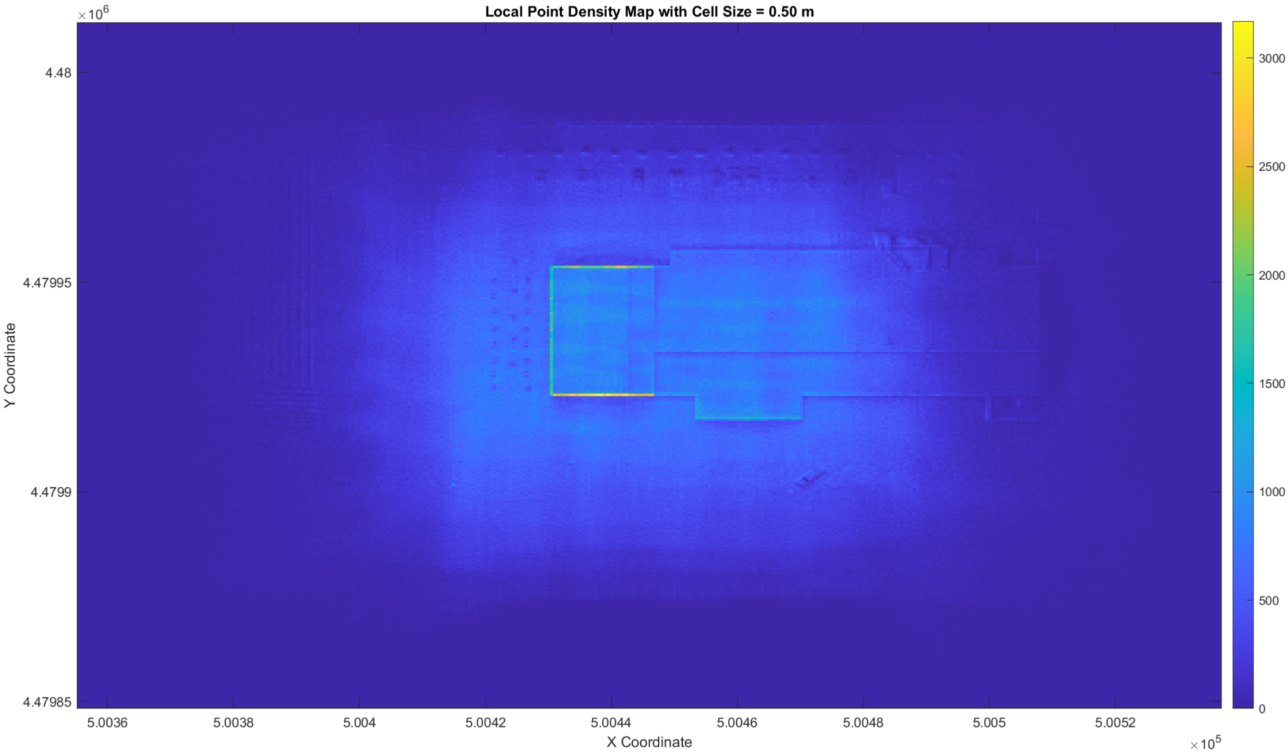

Figure 1: PDM of 20201002_VLP32C_ICSC_ds5.las with a cell size of 0.5 m

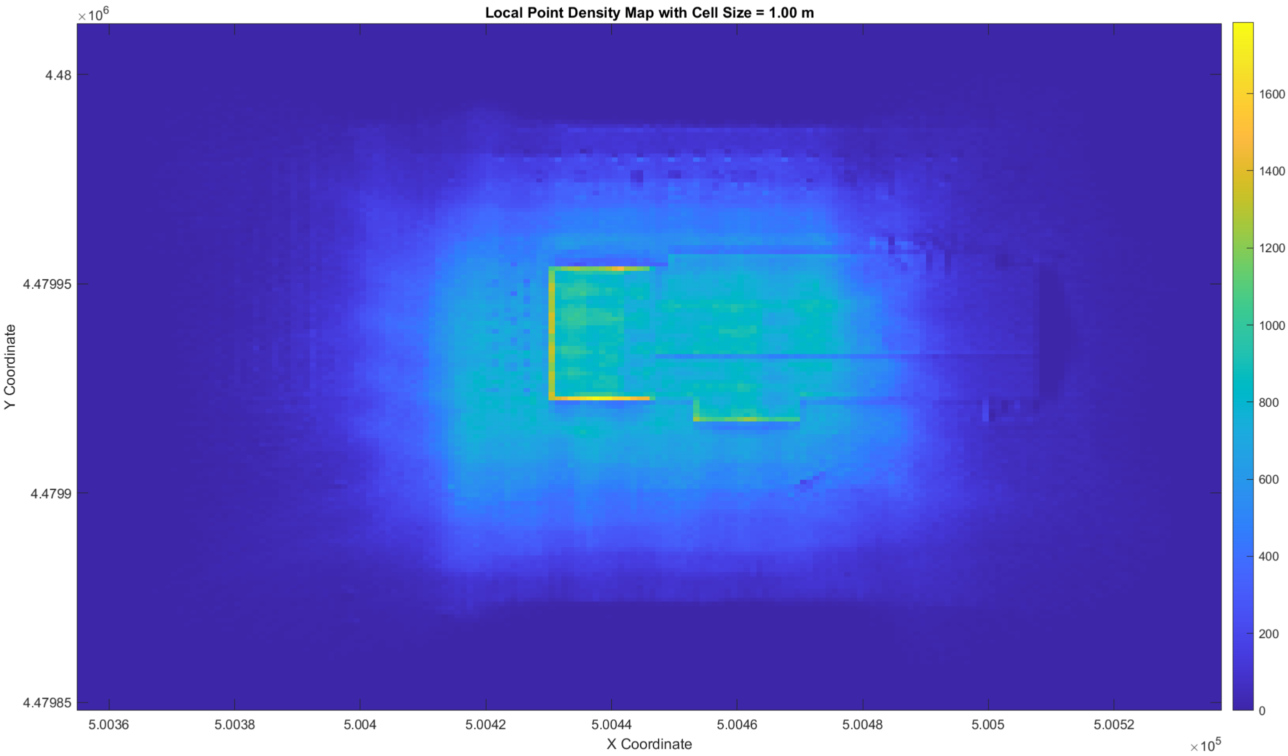

Figure 2: PDM of 20201002_VLP32C_ICSC_ds5.las with a cell size of 1 m

Comparative Observations of 20201002_VLP32C_ICSC_ds5.las

From these observations, we can conclude that the cell size in a point density map greatly affects the level of detail and the ability to detect fine structures or objects. Smaller cells capture more detail but might introduce noise, while larger cells provide a more generalized view of the point density.

In the field of LiDAR data analysis, choosing the appropriate cell size is a trade-off between resolution and noise. For precise applications like feature detection or detailed terrain modeling, smaller cell sizes may be preferred. In contrast, for more general landscape studies or where computational resources are limited, larger cell sizes that provide a quicker, more generalized view may be suitable.

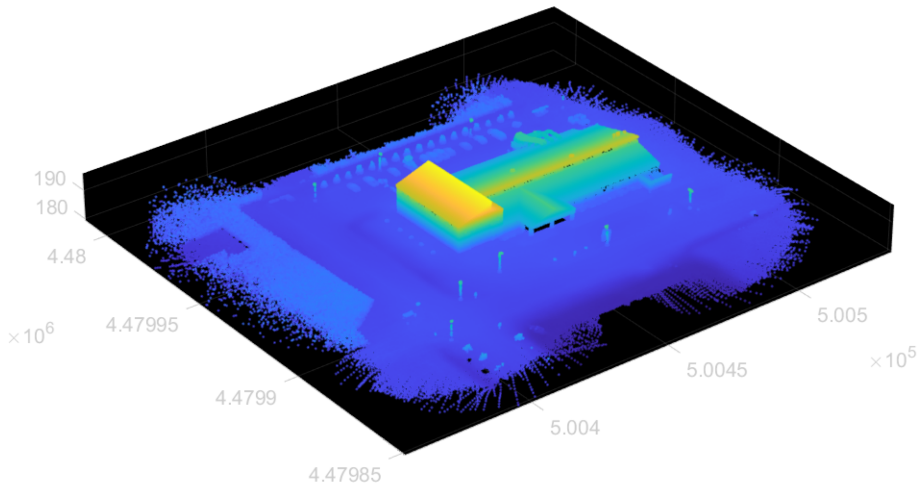

Figure 3: Original point cloud of 20201002_VLP32C_ICSC_ds5.las

```
fileName = 'C:\Users\drsht\Desktop\LIDAR\Assignment2\cloud\20201002_VLP32C_ICSC_ds5.la
lasReader = lasFileReader(fileName);

cell_sizes = [0.5, 1];  % Cell sizes in meters

% Calling the function
point_density_from_las(fileName, cell_sizes);
ptCloud = readPointCloud(lasReader);

fig = figure(Position=[0 0 800 400]);
hPanel = uipanel(fig);
hPlot = axes(hPanel);
pcshow(ptCloud.Location,Parent=hPlot)

disp(lasReader.ClassificationInfo)

function point_density_from_las(filename, cell_sizes)
    % Load the LAS file using lasFileReader
    lasReader = lasFileReader(filename);

    % Extract point cloud
    pts = readPointCloud(lasReader);

    % Extract points assuming the point cloud is organized as x, y, z
    points = pts.Location(:, 1:2);
```

```
    % Iterate over each cell size to calculate and visualize point density
    for cell_size = cell_sizes
        % Define the area of interest
        min_x = min(points(:, 1));
        max_x = max(points(:, 1));
        min_y = min(points(:, 2));
        max_y = max(points(:, 2));

        % Calculate the number of cells along each dimension
        num_cells_x = ceil((max_x - min_x) / cell_size);
        num_cells_y = ceil((max_y - min_y) / cell_size);

        % Initialize the matrix to store point counts
        point_count = zeros(num_cells_y, num_cells_x);

        % Assign each point to a cell
        for i = 1:size(points, 1)
            x_index = floor((points(i, 1) - min_x) / cell_size) + 1;
            y_index = floor((points(i, 2) - min_y) / cell_size) + 1;
            point_count(y_index, x_index) = point_count(y_index, x_index) + 1;
        end

        % Calculate local point density for each cell
        LPD = point_count / cell_size^2;

        % Visualize the local point density
        figure;
        imagesc([min_x max_x], [min_y max_y], LPD);
        colorbar;
        axis xy;
        xlabel('X Coordinate');
        ylabel('Y Coordinate');
        title(sprintf('Local Point Density Map with Cell Size = %.2f m', cell_size));
    end
end
```

For file: 20200827_Riegl_Calib_ds5.las

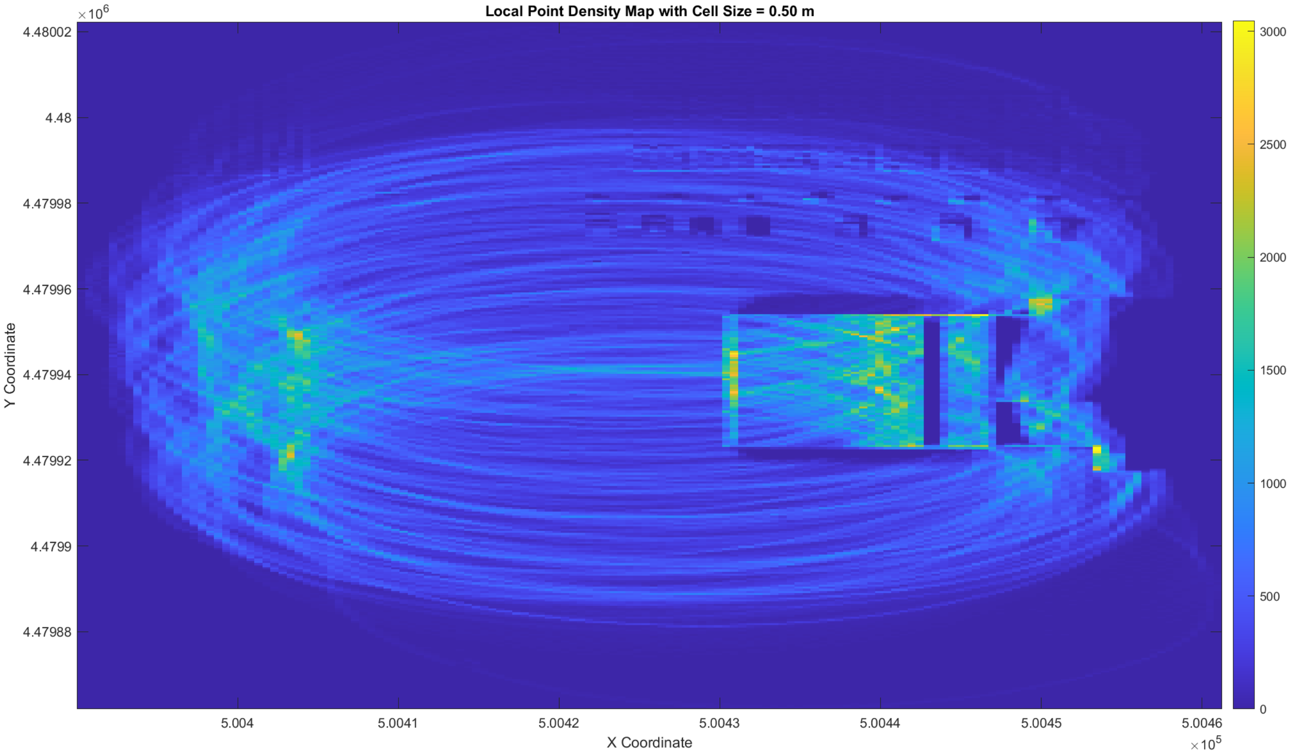

Figure 4: PDM of 20200827_Riegl_Calib_ds5.las with a cell size of 0.5 m

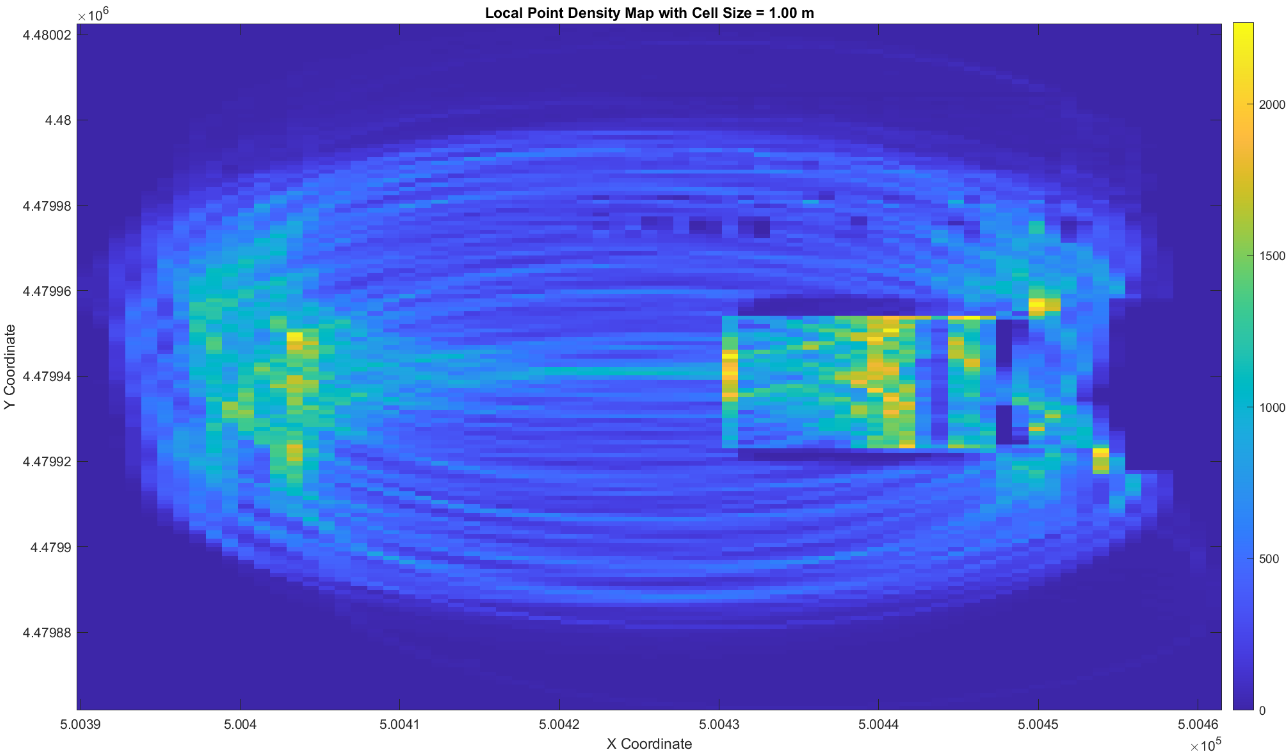

Figure 5: PDM of 20200827_Riegl_Calib_ds5.las with a cell size of 1 m

Comparative Observations of 20200827_Riegl_Calib_ds5.las

By comparing the images side by side, we can draw a few more specific observations:

- Consistency Across Cell Sizes: Certain high-density areas remain prominent in both cell sizes, suggesting these features are significant in the landscape.
- Impact of Resolution on Interpretation: The ability to interpret specific environmental features, such as buildings, trees, or pathways, is enhanced at the higher resolution provided by the 0.5m cell size. In contrast, the 1m cell size abstracts these details but may provide a better overview of the distribution of point densities across the area.
- Choice of Cell Size for Analysis: The selection of cell size has clear implications for the type of analysis that can be conducted. Higher resolutions can support more detailed spatial analysis, while lower resolutions might be sufficient for broader landscape analyses or when computational efficiency is a concern.

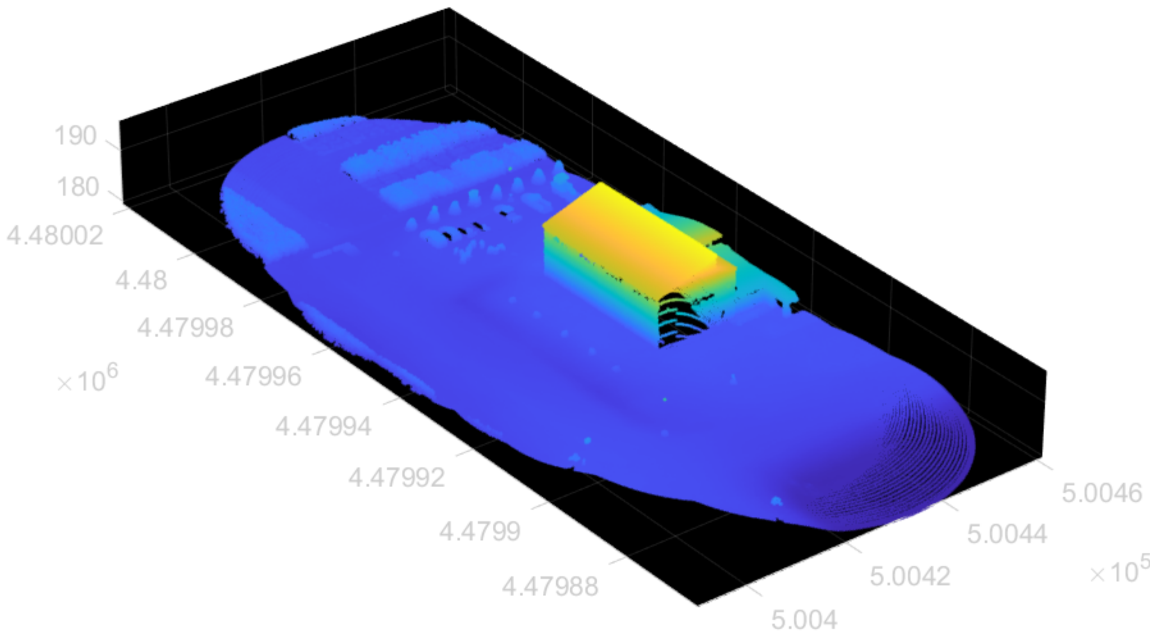

Figure 6: Original point cloud of 20200827_Riegl_Calib_ds5.las

```
fileName = 'C:\Users\drsht\Desktop\LIDAR\Assignment2\cloud\20200827_Riegl_Calib_ds5.l
lasReader = lasFileReader(fileName);

cell_sizes = [0.5, 1];  % Cell sizes in meters

point_density_from_las(fileName, cell_sizes);
ptCloud = readPointCloud(lasReader);

fig = figure(Position=[0 0 800 400]);
hPanel = uipanel(fig);
hPlot = axes(hPanel);
pcshow(ptCloud.Location,Parent=hPlot)

disp(lasReader.ClassificationInfo)

function point_density_from_las(filename, cell_sizes)
    % Load the LAS file using lasFileReader
    lasReader = lasFileReader(filename);

    % Extract point cloud
    pts = readPointCloud(lasReader);
```

```
    % Extract points assuming the point cloud is organized as x, y, z
    points = pts.Location(:, 1:2);

    % Iterate over each cell size to calculate and visualize point density
    for cell_size = cell_sizes
        % Define the area of interest
        min_x = min(points(:, 1));
        max_x = max(points(:, 1));
        min_y = min(points(:, 2));
        max_y = max(points(:, 2));

        % Calculate the number of cells along each dimension
        num_cells_x = ceil((max_x - min_x) / cell_size);
        num_cells_y = ceil((max_y - min_y) / cell_size);

        % Initialize the matrix to store point counts
        point_count = zeros(num_cells_y, num_cells_x);

        % Assign each point to a cell
        for i = 1:size(points, 1)
            x_index = floor((points(i, 1) - min_x) / cell_size) + 1;
            y_index = floor((points(i, 2) - min_y) / cell_size) + 1;
            point_count(y_index, x_index) = point_count(y_index, x_index) + 1;
        end

        % Calculate local point density for each cell
        LPD = point_count / cell_size^2;

        % Visualize the local point density
        figure;
        imagesc([min_x max_x], [min_y max_y], LPD);
        colorbar;
        axis xy;
        xlabel('X Coordinate');
        ylabel('Y Coordinate');
        title(sprintf('Local Point Density Map with Cell Size = %.2f m', cell_size));
    end
end
```