Лабораторна робота №7

Визначення об’єму фігури з хмари точок за допомогою Matlab

7.1 Мета роботи

Ознайомлення з можливостями цифрового обчислення об’єму фігури з хмари точок за допомогою MatLab ..

7.2 Основні теоретичні відомості

Хмара точок (англ. point cloud) — набір даних про точки в деякій системі координат.

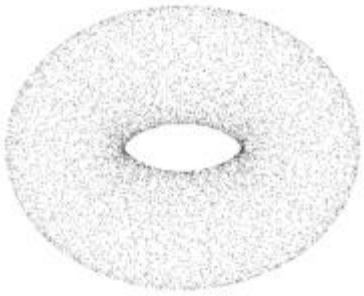

Рис.7.1. - Хмара точок зображення тора.

Рис7.2. - Хмара точок породжена гео-даними парку Ред Рокс, які отримані з дрона.

У тривимірній системі координат, точки визначаються координатами X, Y та Z, і часто призначаються для представлення зовнішньої поверхні об'єкта.

Хмара точок може бути створена 3D-сканером. Такі пристрої породжують дуже велику кількість точок поверхні об'єкту і часто результатом роботи є хмара точок у вигляді файлу даних. В цьому випадку, хмара точок представляє собою множину точок, отриману при скануванні поверхні.

Результати роботи 3D-сканерів — хмари точок — використовуються з різною метою, зокрема для створення 3D САПР моделей для виробництва деталей, метрології/контролю якості та для використання у чисельних застосунках візуалізації, анімації, рендерінгу та масового індивідуалізованого виробництва.

Хоча хмари точок можуть бути безпосередньо відрендерені та перевірені, проте зазвичай не придатні для безпосереднього використання у більшості 3D-застосунків. Тому вони, як правило,

перетворюються в моделі з полігональною або трикутною сіткою, у NURBS модель або САПР модель за допомогою процесу так званої реконструкції поверхні.

Існує багато методів перетворення хмари точок у 3D поверхню. Зокрема, у статті М. Берже і групи авторів наведена порівняльна таблиця з 35 методів відтворення поверхні з хмари точок. Деякі наближення, такі як тріангуляція Делоне, Альфа-форма та метод поворотних куль, будують трикутну або полігональну сітку по вже наявним вершинам хмари точок, а інші наближення будують об'ємні таблиці відстаней або реконструюють неявну поверхню за допомогою алгоритму крокуючих кубиків .

Одним з застосунків, де хмари точок використовуються безпосередньо, є індустріальна метрологія та перевірка якості з використанням промислової комп'ютерної томографії . Хмара точок, отримана в результаті тривимірного сканування готового промислового виробу, може бути приведена у відповідність з CAD-моделлю цього виробу або навіть іншої хмарі точок, і в результаті порівняння можна виявити відмінності між проектними і фактичними параметрами. Ці відмінності можуть відображатися у вигляді кольорових карт, на яких місця і ділянки відхилень між фактичною і формальною моделлю можуть бути автоматично виділеними певним індикатором. Геометричні розміри та допуски також можуть бути безпосередньо отримані з хмари точок.

Хмари точок можуть використовуватися для представлення і візуалізації об'ємних даних, наприклад, в галузі медичної візуалізації. Завдяки використанню хмар точок в таких задачах досягається мультісемплінг і стиснення даних.

У геоінформаційних системах, хмари точок є одним з джерел, які використовуються для створення цифрової моделі рельєфу місцевості. Хмари точок також використовуються для створення цифрових моделей міської місцевості.

Програмна технологія під назвою «Unlimited Detail», яка використовує хмари точок для рендеринга в реальному часі, розробляється австралійською компанією Euclideonз 2003 року.

Етапи обчислення об'єму з хмари точок

Для обчислення об’єму з хмари точок у MATLAB можна використовувати різні підходи. Основна ідея полягає у визначенні замкненої поверхні, що описує об’єкт, та обчисленні об’єму простору, обмеженого цією поверхнею. Етапи роботи:

Попередня обробка хмари точок:

- Фільтрація шуму: Видалення зайвих точок, спричинених похибками сенсора.
- Розрідження (downsampling): Зменшення кількості точок для оптимізації обчислень.
- Виділення області інтересу: Вибір лише тих точок, які належать до об'єкта, обсяг якого потрібно обчислити.

Триангуляція:

Для побудови поверхні об’єкта використовують алгоритм триангуляції, наприклад, Delaunay triangulation. MATLAB надає функцію delaunayTriangulation для створення сітки трикутників.

Точки хмари перетворюються у 3D-сітку, яка повинна бути замкненою для правильного обчислення об'єму.

MATLAB має функцію convhull, яка дозволяє знайти опуклу оболонку точки. Об’єм можна обчислити за допомогою функції volume на основі цієї оболонки.

Якщо об’єкт не є опуклим, застосовуються складніші методи, такі як поділ об'єкта на простіші частини або обчислення за допомогою інтегралів.

7.3 Підготовка до роботи

7.3.1 Ознайомитись з теоретичною інформацією;
7.3.2 Ознайомитись з кодом в додатку 1;
7.3.3 Завантажити код який надано в додатку 1 в програмне середовище MatLab;
7.3.4 Виконати поставленні завдання.

7.4 Виконання роботи

7.4.1 Для фігури куб провести обрахунок значення об’єму з різною щільністю (10, 20, 30, 40, 50, 60, 70, 80, 90, 100) генерації точок. Довжина ребра дорівнює номеру варіанту відповідно до списку.
7.4.2 Для фігури сфера провести обрахунок значення об’єму з різною щільністю (25, 50, 75, 100, 125, 150, 175, 200, 225, 250) генерації точок. Радіус дорівнює номеру варіанту відповідно до списку.
7.4.3 Для фігури конус провести обрахунок значення об’єму з різною щільністю (500, 1000, 1500, 2000, 2500, 3000, 3500, 4000, 4500, 5000) генерації точок. Радіус основи дорівнює номеру варіанту відповідно до списку, а висота обраховується за формулою R^2 * 0.3 .

7.5 Зміст звіту

7.5.1 Найменування і мета роботи.
7.5.2 Зображення хмари точок відповідно до кожного значення щільності для куба.
7.5.3 Графік залежності об’єму до значення щільності.
7.5.4 Порівняти теоретичне значення з тими, що вийшли та зробити висновки.
7.5.5 Зображення хмари точок відповідно до кожного значення щільності для сфери.
7.5.6 Графік залежності об’єму до значення щільності.
7.5.7 Порівняти теоретичне значення з тими, що вийшли та зробити висновки.
7.5.8 Зображення хмари точок відповідно до кожного значення щільності для конуса.
7.5.9 Графік залежності об’єму до значення щільності.
7.5.10 Порівняти теоретичне значення з тими, що вийшли та зробити висновки.
7.5.11 Висновки по роботі.

6.6 Контрольні запитання

7.6.1 Що таке хмара точок, і як вона зазвичай отримується?
7.6.2 Які основні етапи обчислення об'єму з хмари точок у MATLAB?
7.6.3 Для чого використовується триангуляція, і яка функція MATLAB застосовується для її виконання?
7.6.4 Що таке опукла оболонка, і як вона використовується для обчислення об'єму хмари точок?
7.6.5 Які методи можна застосувати для фільтрації шуму в хмарі точок?
7.6.6 Чому важливо, щоб поверхня, побудована з хмари точок, була замкненою для обчислення об’єму?
7.6.7 Яка математична формула використовується для обчислення об'єму багатогранника, побудованого з хмари точок?

```
% ########################################################

% Дослідження куба
% edge_length = 10 ; % Довжина сторони куба
% density = 100;    % Щільність точок
% createCubeCloud(edge_length, density)
% ########################################################
% Дослідження сфери
% radius = 15;
% density = 250;
% createSpherePointCloudWithVolume(radius, density)
% ########################################################
% Дослідження конусу
% R = 20; % радіус основи
% h = R^2*0.3; % висота конуса
% density = 50000; % кількість точок
% createConusCloud(R, h, density);
% ########################################################

function createCubeCloud(edge_length, density)
    % Розрахунок кроку
    step = edge_length / density;

    % Генерація сітки
    x = 0:step:edge_length;
    y = 0:step:edge_length;
    z = 0:step:edge_length;

    % Точки на кожній грані куба
    points = [];

    % Передня та задня грані
    [X, Y] = meshgrid(x, y);
    points = [points; [X(:), Y(:), zeros(size(X(:)))]]; % z = 0
    points = [points; [X(:), Y(:), edge_length * ones(size(X(:)))]]; % z = edge_length

    % Ліва та права грані
    [Y, Z] = meshgrid(y, z);
    points = [points; [zeros(size(Y(:))), Y(:), Z(:)]]; % x = 0
    points = [points; [edge_length * ones(size(Y(:))), Y(:), Z(:)]]; % x = edge_length

    % Верхня та нижня грані
    [X, Z] = meshgrid(x, z);
    points = [points; [X(:), zeros(size(X(:))), Z(:)]]; % y = 0
    points = [points; [X(:), edge_length * ones(size(X(:))), Z(:)]]; % y = edge_length

    % Візуалізація
    scatter3(points(:,1), points(:,2), points(:,3));
    xlabel('X');
    ylabel('Y');
    zlabel('Z');
    title('Cube Point Cloud');
    axis equal;
    % Розрахунок опуклої оболонки та об'єму
    [K, volume] = convhull(points);
```

```
    disp(['Об’єм фігури (куба) визначений программно: ', num2str(volume), ' кубічних
одиниць']);

    theoretical_volume = edge_length^3;
    disp(['Теоретичний об’єм куба: ', num2str(theoretical_volume), ' кубічних одиниць']);
end
function createSpherePointCloudWithVolume(radius, density)
    if nargin < 3
        filename = 'sphere_point_cloud.ply'; % Default filename
    end

    % Generate spherical coordinates
    theta = linspace(0, 2 * pi, density); % Azimuthal angle
    phi = linspace(0, pi, density); % Polar angle

    [Theta, Phi] = meshgrid(theta, phi);

    % Convert spherical coordinates to Cartesian coordinates
    X = radius * sin(Phi) .* cos(Theta);
    Y = radius * sin(Phi) .* sin(Theta);
    Z = radius * cos(Phi);

    % Reshape points into a single matrix
    points = [X(:), Y(:), Z(:)];

    % Visualize the point cloud
    scatter3(points(:,1), points(:,2), points(:,3));
    xlabel('X');
    ylabel('Y');
    zlabel('Z');
    title('Sphere Point Cloud');

    axis equal;

    % Calculate the convex hull volume
    [K, volume] = convhull(points);
    disp(['Об’єм фігури (сфери) визначений программно: ', num2str(volume), ' кубічних
одиниць']);

    % Theoretical volume of the sphere for verification
    theoretical_volume = (4 / 3) * pi * radius^3;
    disp(['Теоретичний об’єм сфери: ', num2str(theoretical_volume), ' кубічних одиниць']);
end

function createConusCloud(R,h,num_points)
    % Генерація рівномірно розподілених точок по висоті
    z = linspace(0, h, num_points)'; % висоти точок
    r = (R * (h - z)) / h; % радіус на кожному рівні залежить від висоти

    % Генерація випадкових кутів для кожної точки
    theta = 2 * pi * rand(num_points, 1); % випадкові кути

    % Перетворення полярних координат у декартові
    x = r .* cos(theta);
    y = r .* sin(theta);

    points = [x(:), y(:), z(:)];
```

```
    % Візуалізація
    scatter3(points(:,1), points(:,2), points(:,3));
    xlabel('X');
    ylabel('Y');
    zlabel('Z');
    title('Cube Point Cloud');
    axis equal;
    % Розрахунок опуклої оболонки та об'єму
    [K, volume] = convhull(points);
    disp(['Об’єм фігури (куба) визначений программно: ', num2str(volume), ' кубічних
одиниць']);

    theoretical_volume = 1/3* pi * R^2 * h ;
    disp(['Теоретичний об’єм куба: ', num2str(theoretical_volume), ' кубічних одиниць']);
end
```