地面と球の影の計算

概要

本ドキュメントでは、光源と球の影が地面に投影されるシミュレーションを MATLAB で実装する方法を示します。地面は離散的なデータとして与えられ、影の計算には光線追跡法と球の交差判定を使用します。影は地面のカラーデータとして表現されます。

理論的背景

影の計算には次の 2 つの主要なステップがあります。

1. 光線のモデル

光源 L から球の表面上の任意の点 P への光線を定義します。この光線のベクトルは以下の式で表されます。

$$\mathbf{v} = \mathbf{P} - \mathbf{L}$$

ここで、$\mathbf{P}$ は球上の点、$\mathbf{L}$ は光源の位置を表します。

2. 球と光線の交差判定

光線 $\mathbf{v}$ が球と交差するかを判定するために、球の方程式を利用します。球の方程式は以下のように表されます。

$$\|\mathbf{X} - \mathbf{C}\|^2 = R^2$$

ここで、$\mathbf{C}$ は球の中心、R は球の半径、$\mathbf{X}$ は球の任意の点を表します。

光線 $\mathbf{v}$ をパラメータ t を用いて表すと、交差する点は以下の式で求められます。

$$\mathbf{X}(t) = \mathbf{L} + t\mathbf{v}$$

これを球の方程式に代入し、t を解きます。結果は二次方程式の解として得られます。

$$t = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a}$$

ここで、

$$a = \|\mathbf{v}\|^2, \quad b = 2(\mathbf{L} - \mathbf{C}) \cdot \mathbf{v}, \quad c = \|\mathbf{L} - \mathbf{C}\|^2 - R^2$$

3. 地面の色を変更

影が落ちる位置を地面上で計算し、その位置の色データを黒（0）に変更します。これにより、地面のカラーデータを通じて影を可視化します。

MATLAB コード

以下は MATLAB での実装コードです。

Listing 1: MATLAB コード: 影の計算と可視化

```
clear; clc; close all;

% 光源の位置
L = [0, 0, 10]; % 光源位置

% 地面の離散データ
[X, Y] = meshgrid(-4:0.05:4, -4:0.05:4); % 形式MESHGRID
Z = sin(X) .* cos(Y); % 地面の形状

% MESHGRID -> NDGRID 形式に変換
X = X'; Y = Y'; Z = Z';

% 地面の補間関数
f_interp = griddedInterpolant(X, Y, Z, 'spline', 'none'); % 地面の高さを補間

% 球の定義
sphereCenter = [0, 0, 3]; % 球の中心
sphereRadius = 1; % 球の半径
[XS, YS, ZS] = sphere(50); % 球の離散的な表面
XS = sphereRadius * XS + sphereCenter(1);
YS = sphereRadius * YS + sphereCenter(2);
ZS = sphereRadius * ZS + sphereCenter(3);

% 地面のカラーデータ（初期状態で均一）
C = ones(size(Z)); % 全体を（白）に設定1

% 各地面格子点について影を計算
for i = 1:size(X, 1)
    for j = 1:size(X, 2)
        % 地面の現在の格子点
        groundPoint = [X(i, j), Y(i, j), Z(i, j)];

        % 光線を地面の点から逆にトレース
        v = L - groundPoint;

        % 球との交差判定
        t = checkSphereIntersection(sphereCenter, sphereRadius,
            groundPoint, v);
        if t > 0 % 交差していれば影を設定
            C(i, j) = 0; % 影の部分を黒に設定
        end
    end
end

% プロット設定
figure;
hold on; grid on; axis equal;
xlabel('X'); ylabel('Y'); zlabel('Z');
view(3);

% 地面をプロット（色データを利用）
surf(X', Y', Z', C', 'FaceAlpha', 0.9, 'EdgeColor', 'none', ...
    'FaceColor', 'interp', 'FaceLighting', 'gouraud', ...
```

```
    'SpecularStrength', 0.5, 'DiffuseStrength', 0.8);

% 球をプロット
surf(XS, YS, ZS, 'EdgeColor', 'none', ...
    'FaceColor', 'blue', 'FaceAlpha', 0.8, 'FaceLighting', 'gouraud', ...
    'SpecularStrength', 0.5, 'DiffuseStrength', 0.8);

% 光源をプロット
plot3(L(1), L(2), L(3), 'yo', 'MarkerSize', 10, 'MarkerFaceColor', 'yellow', 'DisplayName', '光源 L');

% 光源を追加
light('Position', [10 10 10], 'Style', 'infinite'); % 遠方光源
light('Position', [-10 -10 10], 'Style', 'local');   % 局所光源

% 材質設定
material('shiny'); % 表面の光沢を強調

legend show;

%% 関数: 球との交差判定
function t = checkSphereIntersection(center, radius, point, direction)
    % 球との交差判定 (t > 0 のとき交差)
    oc = point - center; % 球の中心から点までのベクトル
    a = dot(direction, direction);
    b = 2.0 * dot(oc, direction);
    c = dot(oc, oc) - radius^2;
    discriminant = b^2 - 4*a*c;

    if discriminant < 0
        t = -1; % 交差なし
    else
        t = (-b - sqrt(discriminant)) / (2.0 * a); % 交差距離
    end
end
```

動作説明

1. 地面のデータを離散的に定義します。地面は連続データから補間可能な形で提供されます。2. 球の各頂点から光線をトレースし、影が地面上に落ちる点を計算します。3. 球と光線の交差判定には二次方程式の解を利用します。4. 地面の色データ（カラーデータ）を影の位置に基づいて変更し、影を可視化します。

結果例

以下に影を可視化した結果例を示します。

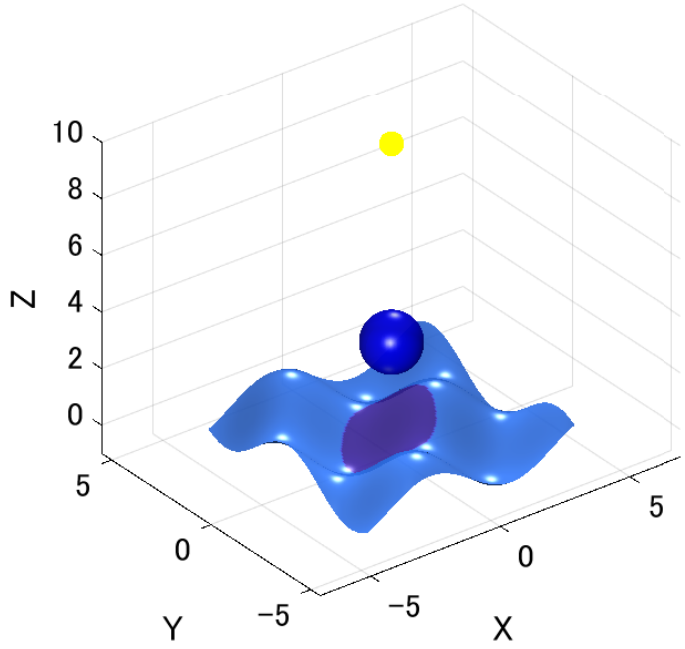

Figure 1: 光源、地面、球、および影の可視化

まとめ

本プログラムは、光源からの光線を利用して影を計算し、地面上の色を動的に変更することで影を可視化する方法を示しました。この手法は球以外の形状や複雑な地形にも応用可能です。