

```

% first step is to load the image and convert it to grayscale
imag = imread('input1.jpg');
if size(imag, 3) == 3
    gray_img = rgb2gray(imag);
else
    gray_img = imag;
end

% Second step is normalizing grayscale
normalizd_img = double(gray_img) / 255;

% Function to clip pixel values to [0, 1] range
clip = @(x) max(0, min(1, x));

% Step 3: Apply Floyd-Steinberg dithering with optimized error diffusion
floyd_steinerg_img = normalizd_img;
[rows, cols] = size(floyd_steinerg_img);

for y = 1:rows
    for x = 1:cols
        old_pixel = floyd_steinerg_img(y, x);
        new_pixel = round(old_pixel);
        floyd_steinerg_img(y, x) = new_pixel;
        quant_error = old_pixel - new_pixel;

        % Distribute error using Floyd-Steinberg kernel (boundary-safe)
        if x + 1 <= cols
            floyd_steinerg_img(y, x + 1) = clip(floyd_steinerg_img(y, x + 1) +
quant_error * 7/16);
        end
        if y + 1 <= rows
            if x - 1 >= 1
                floyd_steinerg_img(y + 1, x - 1) = clip(floyd_steinerg_img(y + 1, x
- 1) + quant_error * 3/16);
            end
            floyd_steinerg_img(y + 1, x) = clip(floyd_steinerg_img(y + 1, x) +
quant_error * 5/16);
            if x + 1 <= cols
                floyd_steinerg_img(y + 1, x + 1) = clip(floyd_steinerg_img(y + 1, x
+ 1) + quant_error * 1/16);
            end
        end
    end
end

% Step 4: Apply Jarvis-Judice-Ninke dithering with optimized error diffusion
jarvis_img = normalizd_img;
J_kernel = [0 0 0 7 5; 3 5 7 5 3; 1 3 5 3 1] / 48;

for y = 1:rows

```

```

for x = 1:cols
    old_pixel = jarvis_img(y, x);
    new_pixel = round(old_pixel);
    jarvis_img(y, x) = new_pixel;
    quant_error = old_pixel - new_pixel;

    % Distribute error using Jarvis-Judice-Ninke kernel (boundary-safe)
    for ky = 1:3
        for kx = -2:2
            ny = y + (ky - 1);
            nx = x + kx;
            if ny <= rows && nx >= 1 && nx <= cols
                jarvis_img(ny, nx) = clip(jarvis_img(ny, nx) + quant_error *
J_kernel(ky, kx + 3));
            end
        end
    end
end

% Step 5: Convert both dithered images back to 8-bit grayscale
floyd_steinerg_img = uint8(floyd_steinerg_img * 255);
jarvis_img = uint8(jarvis_img * 255);

% Step 6: Display the results side by side for comparison
figure;
imshow(gray_img);
title('Original Grayscale Image');

```

Original Grayscale Image



```
figure;  
imshow(floyd_steinerg_img);  
title('Floyd Steinberg Dithering');
```

Floyd Steinberg Dithering



```
figure;  
imshow(jarvis_img);  
title('Jarvis Judice-Ninke Dithering ');
```

