```
% The Kuwahara filter is a nonlinear edge-preserving smoothing filter
% that divides the area surrounding each pixel into several overlapping square
regions.
% For each region, it calculates the mean and variance of the pixel values.
% The filter replaces the pixel value with the mean of the region that has the
lowest variance.
% This approach smooths the image while preserving edges, making it ideal for noise
reduction without excessive blurring of edges.
% Loading the image from the specified path and converting it to grayscale
img = imread('C:\Users\USER\Desktop\Assignment Image.JPG');
gray_img = rgb2gray(img);
gray_img = double(gray_img);  % Converting to double for more precision

% Custom Kuwahara filter function
function output = kuwahara(img, radius)
    % Calculating the window size based on the radius
    window_size = 2 * radius + 1;

    % Padding the image to handle edges
    padded_img = padarray(img, [radius, radius], 'symmetric');
    [rows, cols] = size(img);
    output = zeros(rows, cols);  % Initializing the output image

    % Looping over each pixel in the original image
    for r = 1:rows
        for c = 1:cols
            % Extracting the four quadrants around the pixel
            Q1 = padded_img(r:r+radius, c:c+radius);  % Top-left quadrant
            Q2 = padded_img(r:r+radius, c+radius+1:c+2*radius);  % Top-right
quadrant
            Q3 = padded_img(r+radius+1:r+2*radius, c:c+radius);  % Bottom-left
quadrant
            Q4 = padded_img(r+radius+1:r+2*radius, c+radius+1:c+2*radius);  %
Bottom-right quadrant

            % Calculating mean and variance for each quadrant
            mean1 = mean(Q1(:)); var1 = var(Q1(:));
            mean2 = mean(Q2(:)); var2 = var(Q2(:));
            mean3 = mean(Q3(:)); var3 = var(Q3(:));
            mean4 = mean(Q4(:)); var4 = var(Q4(:));

            % Identifying the quadrant with the lowest variance
            [~, idx] = min([var1, var2, var3, var4]);

            % Assigning the mean of the selected region to the output pixel
            switch idx
                case 1
                    output(r, c) = mean1;
                case 2
                    output(r, c) = mean2;
                case 3
                    output(r, c) = mean3;
                case 4
                    output(r, c) = mean4;
            end
        end
    end
    output = uint8(output);  % Converting the output to uint8 for display
end

% Applying Kuwahara filter with a radius of 3 (7x7 window)
filtered_img = kuwahara(gray_img, 3);

% Saving and displaying the filtered image
imwrite(filtered_img, 'C:\Users\USER\Desktop\kuwahara_filtered_alternative.png');
figure, imshow(filtered_img), title('Kuwahara Filtered Image (Alternative)');
```

Kuwahara Filtered Image (Alternative)