Laplacian Blending: Laplacian pyramid

Next, you will implement the Laplacian pyramid. Remember that every level of the Laplacian comes from taking the same level of the Gaussian pyramid and subtracting the expanded version of the next level. The only exception is the last level which is the same as the last level of the Gaussian pyramid. You will need to use the **gausspyr**, **expand** and **reduce** functions you already implemented.

Your Script

Save Reset MATLAB Documentation (https://www.mathworks.com/help/)

```
% loading the image
A = im2double(imread('orange.png'));
% depth of the pyramids
depth = 5;

% we build the Gaussian pyramid
LA = laplacianpyr(A,depth);

function L = laplacianpyr(I,depth)

    % Input:
    % I: the input image
    % depth: number of levels of the Laplacian pyramid
    % Output:
    % L: a cell containing all the levels of the Laplacian pyramid

    % Please follow the instructions to fill in the missing commands.

    L = cell(1,depth);

    % 1) Create a Gaussian pyramid
    % Use the function you already created.
    G = gausspyr(I,depth);


    % 2) Create a pyramid, where each level is the corresponding level of
    % the Gaussian pyramid minus the expanded version of the next level of
    % the Gaussian pyramid.
    % Remember that the last level of the Laplacian pyramid is the same as
    % the last level of the Gaussian pyramid.
    for i = 1:depth
        if i < depth
            % same level of Gaussian pyramid minus the expanded version of next level
            L{i} = G{i} - expand(G{i+1});
        else
            % same level of Gaussian pyramid
            L{i} = G{i};
        end
    end

end

function G = gausspyr(I,depth)

    % Add your code from the previous step
    G = cell(1,depth);

    % 1) Create a pyramid, where the first level is the original image
    % and every subsequent level is the reduced version of the previous level
    for i = 1:depth
        if i == 1
            G{i} = I; % original image
        else
            G{i} = reduce(G{i-1}); % reduced version of the previous level
        end
    end

end

function g = reduce(I)

    % Add your code from the previous step
    Gauss = fspecial('gaussian',5,1);

    % 2) Convolve the input image with the filter kernel (MATLAB command imfilter)
    % Tip: Use the default settings of imfilter
    I = im2double(I);
    im_filtered = imfilter(I,Gauss);

    % 3) Subsample the image by a factor of 2
    % i.e., keep only 1st, 3rd, 5th, .. rows and columns
    g = im_filtered(1:2:end, 1:2:end,:);

end

function g = expand(I)

    % Add your code from the previous step
    I = im2double(I);
    [m,n,clr] = size(I);
    I_exp = zeros(2*m, 2*n, clr);
    % note: 1:2 gives odd indices
    I_exp(1:2:2*m, 1:2:2*n,:) = I(1:m, 1:n,:);

    % 2) Create a Gaussian kernel of size 5x5 and
    % standard deviation equal to 1 (MATLAB command fspecial)
    Gauss = fspecial('gaussian',5,1);

    % 3) Convolve the input image with the filter kernel (MATLAB command imfilter)
    % Tip: Use the default settings of imfilter
    % Remember to multiply the output of the filtering with a factor of 4
    g = 4*imfilter(I_exp,Gauss);

end
```

▶ Run Script ?

Previous Assessment: All Tests Passed

Submit ?

✓ **Is the estimated output correct?**

Output

```
Code ran without output.
```