Laplacian Blending: Combine pyramids

The next operation you will implement is **combine**, which will create a combined pyramid LS from the two pyramids LA and LB using the nodes of GR as weights. Effectively, for every level d of the pyramid, and every pixel (i, j), the combined pyramid will be given by the equation:

$$LS(d, i, j) = GR(d, i, j) * LA(d, i, j) + (1 - GR(d, i, j)) * LB(d, i, j)$$

Follow the instructions of the script below to complete the **combine** function. Again, the functions you implemented in the previous steps will be useful here.

Your Script

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

```
% we load the two images we will blend
A = im2double(imread('orange.png'));
B = im2double(imread('apple.png'));

% mask that defines the blending region
R = zeros(512,512); R(:,257:512)=1;

% depth of the pyramids
depth = 5;

% 1) we build the Laplacian pyramids of the two images
LA = laplacianpyr(A,depth);
LB = laplacianpyr(B,depth);

% 2) we build the Gaussian pyramid of the selected region
GR = gausspyr(R,depth);

% 3) we combine the two pyramids using the nodes of GR as weights
[LS] = combine(LA, LB, GR);


function [LS] = combine(LA, LB, GR)

    % Input:
    % LA: the Laplacian pyramid of the first image
    % LB: the Laplacian pyramid of the second image
    % GR: Gaussian pyramid of the selected region
    % Output:
    % LS: Combined Laplacian pyramid

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

    depth = numel(LA);
    LS = cell(1,depth);

    % 1) Combine the Laplacian pyramids of the two images.
    % For every level d, and every pixel (i,j) the output for the
    % combined Laplacian pyramid is of the form:
    % LS(d,i,j) = GR(d,i,j)*LA(d,i,j) + (1-GR(d,i,j))*LB(d,i,j)
    for i = 1:depth
        % Put your code here
      [m,n,clr] = size(LB{i});
      one_matrix = ones(m,n,clr);

      LS{i} = GR{i} .* LA{i} + (one_matrix - GR{i}) .* LB{i};
    end
end

function L = laplacianpyr(I,depth)
```

```
    % Add your code from the previous step
    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)
```

```
118     % Standard deviation equal to 1 (MATLAB command fspecial)
119     Gauss = fspecial('gaussian',5,1);
120
121     % 3) Convolve the input image with the filter kernel (MATLAB command imfilter)
122     % Tip: Use the default settings of imfilter
123     % Remember to multiply the output of the filtering with a factor of 4
124     g = 4*imfilter(I_exp,Gauss);
125
126 end
```

Run Script

Previous Assessment: All Tests Passed

Submit

Is the estimated output correct?

Output

```
Code ran without output.
```