```
function B = bilinearInterpl(A, k)

    % recursion for colored images
    if (size(A,3) == 3)
        B = uint8(zeros(size(A,1)*k, size(A,2)*k,3));
        B(:,:,1) = bilinearInterpl(A(:,:,1), k);
        B(:,:,2) = bilinearInterpl(A(:,:,2), k);
        B(:,:,3) = bilinearInterpl(A(:,:,3), k);
        return
    end

    B = uint8(zeros(size(A,1)*k, size(A,2)*k));
    H = size(A,1);
    W = size(A,2);
    kH = k*H;
    kW = k*W;

    % mapping function
    for i = 1 : H
        for j = 1 : W
            B(k*i,k*j) = A(i,j);
        end
    end

    % rows
    for i = k : kH
        leftVal = 0;
        rightVal = 0;
        % finds row with mapped pixels
        if (mod(i,k) == 0)
            for j = k : kW
                % finds mapped pixels
                if (mod(j,k) == 0)
                    leftVal = j;
                    rightVal = j + k;
                else
                   distance = ( double(B(i, rightVal)) - double(B(i, leftVal)) )/ k;
                   B(i, j) = uint8(floor(double(B(i ,(j-1))) + distance));
                end
            end
        end
    end

    % columns
    for j = k : kW
        topVal = 0;
        bottomVal = 0;
        for i = k : kH
            % finds completed rows from linear interpl (row loop)
            if (mod(i,k) == 0)
                topVal = i;
                bottomVal = i + k;
            else
                distance = ( double(B(bottomVal,j)) - double(B(topVal,j)) )/ k;
                B(i,j) = uint8(floor(double(B((i-1),j)) + distance));
            end
        end
    end

    % padding for top rows
    for i = 1 : (k - 1)
        B(i,:) = B(k,:);
    end

    % padding for left columns
    for j = 1 : (k - 1)
        B(:,j) = B(:,k);
    end

end
```