huffmanDecoder

```
function [ decodedImg ] = huffmanDecoder( binaryVector, CodeBook, height, width)

% This function decodes a single binary vector into a black n white image
% Input: 'binaryVector' is a binary vector formed by concatenated huffman
%       codewords representing, e.g. a Y frame.
%         'codebook' is the generated by huffmanCodebook
%         'width' is the width of the image to be decoded
%         'height' is the height of the image to be decoded
% Output: decodedImage, Black and White, type uint8
```

Code

```
% Pre-allocate imagematrix, as a vector
tempimagevector = zeros(height*width, 1);

% Assign variables for faster computing
codebooklength = length(CodeBook);
binvectorlength = length(binaryVector);


% An index for the imagevector, to assign values into a pre-allocated
% vector for faster computing
pixelpos = 1;

% Loop until all elements are checked
i = 1;
while(i <= binvectorlength)

    % Reset the found variable
    found = 0;

    % Reset the codeword that is checked
    codeword = [];

    % Loop the binaries and add the next binary to the checked codeword if
    % the current doesn't exist in the codebook
    while(found == 0 && i <= binvectorlength)

        % Add the current binary to the checked codeword
        codeword(end+1) = binaryVector(i);

        % Loop the codebook to check the current codeword
        % if it exists, for-loop breaks and found-switch is true which
        % breaks the second while loop, that resets the codeword checked.
        for n=1:codebooklength
            if(isequal(codeword,CodeBook{n,2}))
                tempimagevector(pixelpos,1) = CodeBook{n,1};
                found = 1;
                pixelpos = pixelpos + 1;
                break;
            end
```

```
            end

            % Go to next index even if the codeword has been found or not
            i = i+1;

        end
    end

    % Reshape the vector to the right size as given by the input argument
    decodedImg = reshape(tempimagevector, [height,width]);
    decodedImg = uint8(decodedImg);

    end
```

Published with MATLAB® R2014b