```
function [ imagePatches, height, width ] = patchImage( aBWFrame, patchHeight, patc
```

Introduction of patchImage

This function segments an input image/frame into a number of non-overlapping patches as the output.

```
Input:
1. aBWFrame, is a black and white image or frame to be segmented
2. patchWidth and patchHeight define the width and height of the segmented image p
Output:
imagePatches is a multi-dimiension matrix (a stack of segmented patches)
whose size == patchHeight * patchWidth * numPatch
numPatch is the total number of image patches can be segmented from aBWFrame by th
Width/Height, dimensions of the the frame with padded zeros.

%   Note that! patchImage function should be able to handle the cases when
%   width/height of the input frame is not exactly integer times of
%   patchWidth/patchHeight. This can be done by padding zeros to the
%   aBWFrame OR simply ignore the remainders:
%   i.e. If frame size == 10*10 and both patchWidth/patchHeight == 3
%        you can pad zeros to enlarge it to 12*12 and get 16 patches,
%        OR get 9 patches by ignoring a row and a column in the frame.
%   You can decide the zero-padding method yourself.


%This function padds zeros around the frame to not clip any data.

[height, width] = size(aBWFrame);
neededHeight = ceil(height/patchHeight);
neededWidth = ceil(width/patchWidth);

%Padd zeros:
aBWFrame(height: neededHeight * patchHeight, width:neededWidth * patchWidth) = 0;

rows = neededHeight;
columns = neededWidth;

placeholder = zeros(patchHeight, patchWidth, rows*columns);

index = 1;

for row = 1: rows
    rowEnd = row * patchHeight;
    rowStart = rowEnd - patchHeight + 1;


    for col = 1: columns
        columnEnd = col * patchWidth;
        columnStart = columnEnd - patchWidth +1;

        placeholder(:, :, index) = aBWFrame(rowStart:rowEnd, columnStart:columnEnd
```

```
        index = index + 1;

    end
end

imagePatches = placeholder;
[height, width] = size(aBWFrame);

end
```

Published with MATLAB® R2014b