Exercise 6: Draw Any Image

Load image from file

Load an image to process. View the original image.

```
img = imread('MathWorksLogo.jpg');
imshow(img)
```

Extract line traces from image

Convert the image to grayscale, binarize it, and perform morphological operations on it to shrink objects to thin lines. After each operation, view the current state of the image.

```
img2 = rgb2gray(img);
imshow(img2)
img3 = ~imbinarize(img2,'adaptive','ForegroundPolarity','dark');
imshow(img3)
img4 = bwmorph(img3,'clean');
imshow(img4)
img5 = bwmorph(img4,'thin',inf);
imshow(img5)
```

Extract pixels in order

Use the recursive function `getCoords` to generate a sequential list of pixels from the thin-line image. The pixels are provided in the order in which they are returned by the `bwboundaries` function, which follows the boundary path of objects in a binary image.

```
coordsPix = getCoords(img5);
```

Visualize the extracted pixels using a scatter plot. A line plot would connect discontiuous segments where one line trace ends and the next begins. Note that coordinates are stored in a row-column list, so you need to flip the order of these coordinates for functions that draw x-y data.

```
scatter(coordsPix(:,2),coordsPix(:,1),'.')
axis ij equal
```

Break coordinates list into contiguous segments

Use the coords2segments function to convert the `coordsPix` variable from a single list of pixel coordinates to a cell array where each cell contains a contiguous trace of pixels.

```
segmentsPix = coords2segments(coordsPix);
```

Plot each path from the `segmentsPix` variable in a separate color to show the continuous traces that have been extracted.

```
for ii = 1:numel(segmentsPix)
```

```
    coords = segmentsPix{ii};
    plot(coords(:,2),coords(:,1))
    hold on
end
hold off
axis ij equal
```

Merge connected segments

In the previous plot, there may be some segments whose endpoints were adjacent. These endpoints could be extended or merged to minimize gaps in the drawing and draw the image with fewer separate traces. Connect these discontinuities with the `connectSegments` function.

```
segmentsPix = connectSegments(segmentsPix);
```

Plot the modified version of segmentsPix as before. Note that there are now fewer different segments. Some segments have been merged into a single continuous trace.

```
for ii = 1:numel(segmentsPix)
    coords = segmentsPix{ii};
    plot(coords(:,2),coords(:,1))
    hold on
end
hold off
axis ij equal
```

Store x and y pixel limits

In the previous exercise, you needed the minimum and maximum pixel values to for the processed image so that you could scale the pixel values to meters. Compute `xLimPix` and `yLimPix` from the full list of pixel coordinates `coordsPix` so you can use them later when drawing this image.

```
xLimPix = [min(coordsPix(:,2)) max(coordsPix(:,2))];
yLimPix = [min(coordsPix(:,1)) max(coordsPix(:,1))];
```

Test function imageToPixelSegments

After encapsulating all the above steps in a function, run this function on the original image to verify that it does indeed produce the samme results.

```
[segmentsPix2,xLimPix2,yLimPix2] = imageToPixelSegments(img);
```

Use the `isequal` function to compare the output of the `imageToPixelSegments` function with the variables computed in this script.

```
isequal(segmentsPix,segmentsPix2)
isequal(xLimPix,xLimPix2)
isequal(yLimPix,yLimPix2)
```

Draw image using function created previously

As you did in the previou exercise, hang the robot on the whiteboard, measure the initial string lengths, and use the functions you've written to draw the image.

```
Z_i = initialPosition();
drawImageFromPix(segmentsPix,xLimPix,yLimPix,Z_i)
```

Copyright 2018 The MathWorks, Inc.