Example: Image processing (color maps and data structures) • Computer Science • 20min

Image processing involves loading and manipulating image data to determine properties automatically. This can apply to industrial processes, medical data, artificial intelligence and robot vision, and astronomy.

- At the most basic level, we need to understand how to load and save image data, and how image data are represented internally in MATLAB.

```
img = imread('satellite.png');
size(img)
imshow(img);
```

We can separate out the red, blue, and green channels if we know that certain features show up preferentially in each.

- Display the red channel only.

```
imshow(img(:,:,1))
```

- Now we have to colormap this if we want to see it in that color preferentially. A colormap is a set of color values that are used discretely at levels of the values in an array: in this case, 0–255.

```
redmap = zeros(256,3);
redmap(:,1) = linspace(0,1,256);
figure(1);
imshow(img(:,:,1),redmap);

greenmap=zeros(256,3);
greenmap(:,2)=linspace(0,1,256);
figure(2);
imshow(img(:,:,2),greenmap);

bluemap=zeros(256,3);
bluemap(:,3)=linspace(0,1,256);
figure(3);
imshow(img(:,:,3),bluemap);

figure(4)
subplot(2,2,1), subimage(img)
subplot(2,2,2), subimage(img(:,:,1),redmap)
subplot(2,2,3), subimage(img(:,:,2),greenmap)
subplot(2,2,4), subimage(img(:,:,3),bluemap)
```

This allows you to extract color-dependent qualities of images useful for automatic object detection.

Example: Image processing (edge detection) • Computer Science • 30min

We wish to use to solve image detection problems automatically. For instance, a grouping of carcinoma cells is depicted in `cell.png`. We wish to detect the cells in this image automatically.

- Read in the cell and display it.

```
I=imread('cell.png');
figure, imshow(I), title('original image');
text(size(I,2),size(I,1)+15,'Image courtesy Ed Uthman','FontSize',10,
      'HorizontalAlignment','right');
```

- Detect the edges within the cell. This uses the gradient image and a threshold to create a binary mask of the cell.

```
[~, threshold] = edge(I, 'sobel');
fudgeFactor = 0.8;
BWs = edge(I,'sobel', threshold * fudgeFactor);
figure, imshow(BWs), title('binary gradient mask');
```

- Thicken the lines using the structural element tool (strel) and the image dilation tool (imdilate).

```
se90 = strel('line', 3, 90);
se0 = strel('line', 3, 0);
BWsdil = imdilate(BWs, [se90 se0]);
figure, imshow(BWsdil), title('dilated gradient mask');
```

- Now we fill the interior gaps of the cell.

```
BWdfill = imfill(BWsdil, 'holes');
figure, imshow(BWdfill);
title('binary image with filled holes');
```

- Finally, we smooth the object to remove the noisy disconnected globs from outside the cell.

```
seD = strel('diamond',1);
BWfinal = imerode(BWdfill,seD);
BWfinal = imerode(BWfinal,seD);
figure, imshow(BWfinal), title('segmented image');
```

- Superimpose the edge detection over the original image to see the results of our automatic detection script.

```
BWoutline = bwperim(BWfinal);
Segout = I;
Segout(BWoutline) = 0;
figure, imshow(Segout), title('outlined original image');
```

Modified from Mathworks example at http://www.mathworks.com/help/images/examples/detecting-a-cell-using-image-segmentation.html.