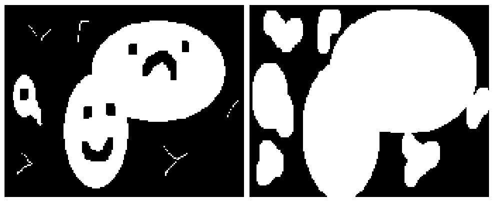
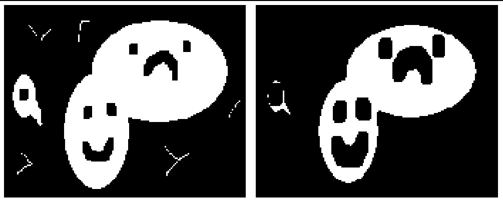
Activity 2: Color and Contrast
Part I: Color Thresholding

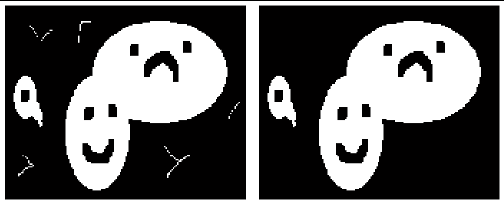
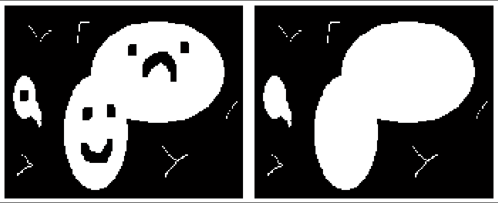
Primary Goal: Learn how to threshold color images interactively.
Secondary Goal: Discuss how to improve your results using binary image operations.

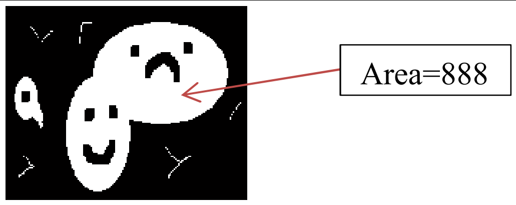
Binary Image Operations

	D (Before) **D2 (After)**

Dilation: Grow the blobs by an amount specified by the structure element. Often we choose a disk for the element. The larger the radius of the disk (r=10 in example below), the more the blobs will grow.
 	
`ball = strel('disk',10);`
`D2 = imdilate(D, ball);`
	

Erosion: Shrink the white regions by an amount specified by the structure element. Note small blobs disappear and holes get larger.
 	
`ball = strel('disk',5);`
`D2 = imerode(D, ball);`
	

Area Open: Remove all blobs that have an area smaller than the specified value. The area threshold depends on the image (A=100 pixels in example below). `D2 = bwareaopen(D,100);`	
Fill: Fill in black regions that are completely surrounded by white pixels. Note this will have no effect on small specks. `D2=imfill(D,'holes');`	

Blob Statistics: The regionprops command can look up useful statistics about the blobs in your image. We could look up the blobs' area, perimeter, centroids, and other geometric features.
 	
`S=regionprops(D,'area');`
`S(1)`
`888`
	

1)) Download the A**ctivity** 2 zip file from the course website to your Desktop. Right-click on the folder and select Extract to unzip the folder.

2.) In Matlab, navigate to the Activity 2 folder and read the image **mm.jpg**.

```
A = imread('mm.jpg');
```

Our goal today is to detect the yellow M&Ms. Display the image using the **imshow** command. Locate a representative yellow pixel and use the Data Cursor tool on the figure to look up the pixel's RGB value. This will be an important reference color for the next step.

YELLOW = (255 , 223 , 0)

3.) In the Activity 2 folder, there is a simple GUI for interactive color thresholding called thresh.m. You can run the program by typing the command **thresh**.

First you need to load a 3-channel RGB image matrix from the workspace. Type A in the text box and press the "Load color image" button next to it. Next adjust the 6 sliders to set threshold bounds for the R, G and B channels. You should see the detection mask form as you move the sliders. It may help to check the "Overlay images" box.

Once you are happy with your detection mask, write down the 6 threshold values you decided upon.

$200 \leq R \leq 255$, $160 \leq G \leq 255$, $1 \leq B \leq 255$

Save the binary image to the variable D by pressing the "Write mask" button.

4.) Try improving the binary image using various binary image operations. For example, you might try to fill in the holes using the **imfill** function. Or you might try to remove specks using the **bwareaopen** function.

5.) Try repeating steps #2-4 to find the red M&Ms. Just be sure not to over-write your final matrix D that you found for the yellow M&Ms, because we will use it in the Part II of today's activity.

Why is it harder to detect the red M&Ms? Edges of Orange smarties fall into the same color threshold.

Activity 2: Color and Contrast
Part II: Evaluating Detection Results

Primary Goal: Learn how to quantitatively evaluate binary detection masks.
Secondary Goal: Discuss TP/TN/FP/FN statistics.

Classification Analysis

Classification Categories

- True Positive (TP): The number of pixels correctly labeled as an object. A TP is also known as hit.
- True Negative (TN): The number of pixels correctly labeled as *not* belonging to an object. A TN is also known as correct rejection.
- False Positive (FP): The number of pixels that were labeled as an object but are actually *not* an object. A FP is also known as false alarm or Type I error.
- False Negative (FN): The number of pixels that were labeled as *not* belonging to an object but are actually an object. A FN is also known as miss or Type II error.

Classification Rates

True Positive Rate (TPR), also known as hit rate, recall, and sensitivity

$$TPR = TP/P = TP/(TP + FN)$$

Specificity (SPC), also known as True Negative Rate

$$SPC = TN/N = TN/(FP + TN)$$

Positive Predictive Value (PPV), also known as precision

$$PPV = TP/(TP + FP)$$

Negative Predictive Value (NPV)

$$NPV = TN/(TN + FN)$$

False Positive Rate (FPR), also known as fall-out

$$FPR = FP/N = FP/(FP + TN)$$

False Discovery Rate (FDR)

$$FDR = FP/(FP + TP) = 1 - PPV$$

Classification Statistics

Accuracy (ACC)

$$ACC = (TP + TN)/(P + N)$$

F1 score

The F1 score is the harmonic mean of precision and sensitivity.

$$F1 = 2TP/(2TP + FP + FN)$$

Matthews Correlation Coefficient (MCC)

MCC takes on values in the range [-1,1]. A value of +1 indicates perfect detection, 0 is no better than random guessing, and -1 means it is completely incorrect.

$$\text{MCC} = \frac{TP \times TN - FP \times FN}{\sqrt{(TP + FP)(TP + FN)(TN + FP)(TN + FN)}}$$

Table adapted from http://en.wikipedia.org/wiki/Precision_and_recall

1.) The image **mm_truth.bmp** contains a hand-labeled mask showing the location of the yellow M&Ms. Load this image into a matrix G.

```
G = imread('mm_truth.bmp');
```

2.) First we will *qualitatively* compare your detection mask D from Part I to the ground truth G. Display your image D next to G in an appropriate subplot. Try to determine visually where the two images differ.

Below is a fun way to overlay two images. What do the colors represent?

```
M(:,:,1)=D;  M(:,:,2)=G;  M(:,:,3)=0;  imagesc(M)
```

3.) Next we will quantitatively compare D and G. We will first count the number of pixels in each of the 4 basic detection categories.

- True Positive (TP): The number of pixels that you labeled as a yellow M&M and the ground truth agrees is a yellow M&M.
- True Positive (TN): The number of pixels that you labeled as *not* belonging to a yellow M&Ms and the ground truth agrees is *not* a yellow M&M.
- False Positive (FP): The number of pixels that you labeled as a yellow M&M but the ground truth says is *not* a yellow M&M.
- False Negative (FN): The number of pixels that you labeled as *not* belonging to a yellow M&M but the ground truth says is actually a yellow M&M.

We can compute these values by summing the 1's in a logical matrix. For example, a false positive (FP) occurs when D=1 but G=0. We can obtain the False Positive count with the command:

```
FP = sum(sum( [D==1 & G==0] ));
```

Note we have to call the **sum** function twice to add up the values in a 2D matrix, because the sum command adds up entries in a 1D vector. The first call sums up the columns and the second sums up the rows.

Compute the 4 counts and write the values in the table below. Your TN value should be much larger than the others.

TP	**TN**	**FP**	**FN**
32501	362012	2308	1929

4.) Finally, let's get a feel for the overall quality of your detection by computing the Matthews' Correlation Coefficient (MCC). The closer your MCC is to 1, the better your result. If you got a negative MCC value, then you did something horribly wrong.

$$\text{MCC} = \frac{TP \times TN - FP \times FN}{\sqrt{(TP+FP)(TP+FN)(TN+FP)(TN+FN)}} = \underline{\quad 0.9330 \quad}$$