Aaron Feinberg
CMPSC-497
Lab 1a - Detecting Circular Objects
Materials: Matlab, Household Objects, Camera (Phone)

Discussion

In this Lab, I made of use of a few of MATLAB's image processing and computer vision features to determine the "roundness" of an object. We first take the raw image and convert it to gray scale and into a binary representation of the image. We then apply a few post processing techniques such as grain and noise removal (removing all groups of pixels below a certain threshold) to avoid having them counted as objects.

Roundness Algorithm

We find the size of each distinct object on the screen and then use the pixels at the boundary of each object to determine its shape and then we calculate its perimeter. We then super-impose a circle over the object. This circle has a radius such that the circle's perimeter would be equal to that of the object it is superimposed over. We then calculate the following ratio of these 2 figure's areas:

$$metric = \frac{a_{original}}{a_{circle}}$$

The close this metric is to 1, the more "round" a given object may be. To keep things interesting (and since i didn't really have too many distinct perfectly round objects), I set the minimum roundness threshold to 0.70.

Code

Since we'd be performing this operation 3 images, i created a function called `processImage()` which accepts 3 parameters.

- `im` - the image
- `roundness_threshold` - the minimum thershold for assuming something is "round" or not
- `graypoint` - a value for which the builtin `rgb2gray()` function should use as the graypoint (-1 for letting the function itself decide).

```
% importing image of objects
im_1 = imread('/home/aaron/Pictures/circular_obj.jpg');
im_2 = imread('/home/aaron/Pictures/circular_obj_2.jpg');
im_3 = imread('/home/aaron/Pictures/circular_obj_3.jpg');

% processing 3 images - adjusting graypoint and roudness threshold
as needed
processImage(im_1, .70, .165)
processImage(im_2, .70, .35)
processImage(im_3, .85, -1)

% input: image, minimum roundness value, gray point (-1 lets algo
choose)
% returns total objects and number of round objects
function [total_objects, round_objects] = processImage(im,
roundness_threshold, graypoint)
    % display original image
   imtool(im);


    % converting to grayscale
    im_gray = rgb2gray(im);


    % binarizing the image (passing -1 lets us have the default
algorithm choose the gray point)
    if graypoint == -1
        graypoint = graythresh(im_gray);
    end

    im_bin_threshold = imbinarize(im_gray, graypoint); % removes
objects
```

```
    % removing noise/grains
    im_bin_denoise = bwareaopen(im_bin_threshold, 1000); % lots of
tweaking

    % filling in any holes in the image
    se = strel('disk', 15); % lots of tweaking
    im_bin_closed = imclose(im_bin_denoise, se);
    im_bin_filled = imfill(im_bin_closed, 'holes');


    % Boundary and Label Matrices
    [B, L] = bwboundaries(im_bin_filled, 'noholes')
    imshow(label2rgb(L, @jet, [.5 .5 .5])) % Displaying the distinct
objects



     % getting true and circular areas
     stats = regionprops(L, 'Area', 'Centroid');

     % counting round objects (those that are above threshold value)
     objects_exceed_threshold = 0;

      % appending more data to the graph
    hold on


    % drawing boundaries
    for k = 1 : length(B)
            boundary = B{k}
            plot(boundary(:,2), boundary(:,1), 'w', 'LineWidth', 2)

        delta_sq = diff(boundary).^2;
        perimeter = sum(sqrt(sum(delta_sq,2)));

        area1 = perimeter^2/(4*pi);
        area2 = stats(k).Area;

        area_true = stats(k).Area;
        metric = area2/area1
```

```
        metric_string = sprintf("%2.2f",metric);

        % plot a circle about the centroid
        if metric > roundness_threshold
                centroid = stats(k).Centroid;
            objects_exceed_threshold = objects_exceed_threshold + 1;
                plot(centroid(1), centroid(2), 'ko');
        end

        % plot the roundness or ciruclarity metric near the object
        text(boundary(1,2)-100, boundary(1,1)+120, metric_string,
'color', 'y','FontSize', 14, 'FontWeight', 'bold');
    end

     hold off

    sprintf("Total Objects: %.f", length(B))
    sprintf("Objects Exceeding Roundness Threshold: %.f",
objects_exceed_threshold)
    total_objects = length(B);
    round_objects = objects_exceed_threshold;

end
```

Results

Image 1

B = 5×1 cell

	1
1	594×2 double
2	1081×2 double
3	261×2 double
4	410×2 double

	1
5	600×2 double

L = 905×464

0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0

0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0

0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0

0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0

0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0

0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0

0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0

0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0

0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0

0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0

boundary = 594×2

494 1

493 2

492 3

491 4

490 5

489 6

488 6

487 7

486 8

485 9

metric = 0.2116

boundary = 1081×2

733 29
732 30
731 30
730 30
729 30
728 31
727 31
726 31
725 31
724 32

metric = 0.7155

boundary = 261×2

136 60
135 61
134 62
133 63
132 64
132 65
131 66
130 67
129 68
129 69

metric = 0.4739

boundary = 410×2

414 225
413 226
412 226
411 227
410 227
409 227
408 228

407 229
406 230
405 231

metric = 0.7549

boundary = 600×2
83 297
82 298
81 298
80 299
79 299
78 299
77 299
76 299
75 300
74 301

metric = 0.2682

ans = "Total Objects: 5"

ans = "Objects Exceeding Roundness Threshold: 2"

ans = 5

Image 2

B = 4×1 cell

	1
1	1416×2 double
2	2793×2 double
3	4016×2 double

	1
4	2310×2 double

L = 2190×2059
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0

boundary = 1416×2
262 189
261 190
260 190
259 190
258 190
257 190
256 190
255 190
254 190
253 190

metric = 0.8430

boundary = 2793×2
2029 256
2028 257
2027 257
2026 257
2025 257
2024 257
2023 257
2022 257
2021 257
2020 257

metric = 0.2290

boundary = 4016×2
1809 855
1808 856
1808 857
1808 858
1808 859
1808 860
1808 861
1808 862
1808 863
1808 864

metric = 0.3936

boundary = 2310×2
432 892
431 893
430 893
429 893
428 894
427 894
426 895

425 895
424 896
423 896

metric = 0.6639

ans = "Total Objects: 4"

ans = "Objects Exceeding Roundness Threshold: 1"

ans = 4

Image 3

B = 4×1 cell

	1
1	3123×2 double
2	1187×2 double
3	493×2 double
4	819×2 double

L = 2024×1706
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0

0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0

boundary = 3123×2
1139 9
1139 10
1139 11
1139 12
1139 13
1139 14
1139 15
1138 16
1137 17
1136 18

metric = 0.5922

boundary = 1187×2
1498 595
1497 596
1496 596
1495 596
1494 596
1493 596
1492 596
1491 597
1490 597
1489 597

metric = 0.8097

boundary = 493×2
819 1230
818 1231
817 1231
816 1232
815 1232
814 1232
813 1232
812 1233
811 1233
810 1233

metric = 0.9037

boundary = 819×2
394 1356
393 1357
392 1357
391 1357
390 1357
389 1357
388 1357
387 1357
386 1358
385 1358

metric = 0.7936

ans = "Total Objects: 4"

ans = "Objects Exceeding Roundness Threshold: 3"

ans = 4

Problems

One of the main problems I encountered was the issue of the default gray point. I found that in order to avoid doing a lot of post processing in MATLAB, I needed to find objects with a lot of contrast between the background. The default gray point that was chosen was often sensible as an average brightness value but that was actually detrimental in most cases for this lab. It was also clear that any sort of glare or uneven lighting showed up as an object when I binarized the image. Also, objects like the little cologne bottle in the first image had a black logo in the center which when binarized created a hole in the center of the bottle. This required me to play with the imclose() and imfill() functions quite a bit (as well as for other objects).

In summary, selecting a high contrasting gray point, allowed you to set a black point such that you wouldn't lose any objects nor pickup a lot of noise and grain.

Question: Can this algorithm be used to count number of M&Ms?

Yes, Since M&Ms are uniformly sized then it seems reasonable. The only difficulty would be to have enough contrast between M&Ms of a darker color and those of a lighter color. It would also be important to fine tune parameters like the bwfill() values are large enough to ensure that the M inside the M&M gets filled but small enough that the entire M&M isn't removed. This would be made easier with consistent lighting and a high contrast, uniform background.