describe your algorithm/ how it works

- extract image from file
  - take R value of image (R proved to have the most contrast)
- detect all edges using ‘canny’
  - Dilate the image, with a strel(‘diamond’, 2)
    - diamond proved to be the most viable shape of all tried. this is because diamond helps connect the edges thus avoiding unconnected lines
  - erode image using strel(‘disk’,1)
    - This helps erode some of the connected edges of the cig that we dont want to connect
    - The above 2 functions basically perform a morphological closing on the image but with different “brushes”, Result is shown in following image

At this point we have the outline of all the objects in the image.

- *“[B,L,N] = bwboundaries(BW1_3);”* AND “*stats= regionprops(L, ‘all’)*;” are then used to exact all the boundaries of all the objects in the image and all the stats are constructed using regionprops

- I use Area, Perimeter, Eccentricity, Solidity, MajorAxisLength, MinorAxisLength to detect a rectangular shape and identify that as a cigarette butt.

- I used weights to assign different weighings to different parameters so that different parameters will have different affect on determining if the object is infact a cigarette butt.

    1 = stats(k).Area;
    2 = stats(k).Perimeter/stats(k).Area;
    3 = stats(k).Eccentricity;
    4 = stats(k).Solidity;
    5 = rect(1,k)/(stats(k).MajorAxisLength*stats(k).MinorAxisLength);
    6 = stats(k).MajorAxisLength/stats(k).MinorAxisLength;

- These are the 6 parameters I used to define my rectangle. I have defined bounds on each of these in my code.

- 1 is important because in a given image, all cigs will roughly be same area. If my algorithm detects 2 objects of very different area, then something is wrong

- 2 is important because a rectangle has a certain P/A ratio

- 3 determines how closely the shape loops to an eclipse (This is the only parameter i had to begin with, then I added the rest 4)

- 4 according to me determines how likely is the shape to exist in real world. Officially defined as: Area/ConvexArea (Proportion of the pixels in the convex hull that are also in the region)

- 5 is important, again, to see how much our object looks like a rectangle. Major*Minor (length*width) gives us a rectangle of the current object. If area/(major*minor) is closer to 1, then it means it is more likely to be a rectangle.

- 6 is important to distinguish long lines from a good porportionate rectangle. i.e. if a pic of sidewalk is presented (also a rectangle) then it will be detected as a cig. 6 limits my length/width ratio to only allow certain l/w rectangles to pass

Once i have scores from all weights, I add them up for each object in rect(8,k); divide rect(8,:) by max(rect(8,k)) to normalize values from [0,1]. And then declare any object that has score of 95%+ to be a cig.

what you tried

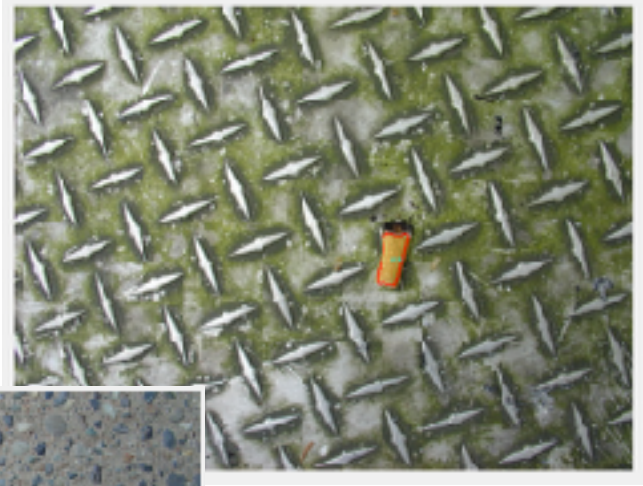
Please find details in journal

why it works

this method works because I am not disregarding any object in the image. I have tried to simplify the image by only keeping the cig in the image but that proved to be very challenging. Multiple morphological processes were required to remove the background and it was not general.

In this latest method, I am keeping all the objects in the image, and then performing analysis on each object to see if the object resembles a cig or not.

few examples are as follows:



its limitations

My code cannot detect multiple cigs if the cigs are vastly different. I.e. If one cig is 100% unused, and another is 75% used, my code might have trouble finding the smaller one.

Another limitation is that it since I was not able to detect very soft edges (orange paper), my algorithm cant detect cigs on smooth surface that blend in with the cig.

Cigs must be the fairly big in the image. ‘Cig01.JPG’ was hard to work with.

I was able to detect ALL cigs atleast with one combination of parameters, but this submission has the most general code. Overall my algorithm is able to detect between 70-95% of cigs depending on background conditions