Assignment A3: Enhancement Filters

CS 4640
Fall 2021

Assigned: 14 September 2021

Due: 30 September 2021

For this problem, handin your lab report and the Matlab .m files for the functions described by the headers below. Describe the details of your appoach, issues that came up, and give example results. Give defect performance results on all 141 test images for each individual defect detector (this does not include the cap boundary extractor).

Some notes:

- Indent headers correctly (5 spaces indented lines)
- Do not exceed 75 characters per source line

None of the functions should write to the interpreter, draw, etc.

1. Develop a function, *CS4640_label_OK*, described by the header below. Create a 3x3 $\frac{\partial f}{\partial y}$ filter and determine if there is a reasonable horizontal edge at the upper label area. Show the filter, as well as example results of the filter processing (e.g., an image of imfilter output running the filter).

```
function b = CS4640_label_OK(im)
% CS4640_label_OK - bottle label is OK on middle bottle in image
% On input:
%     im (MxNx3 array): input RGB image
% On output:
%     b (Boolean): 1 if label is OK; else 0
```

```
% Call:
%     b = CS4640_label_OK(bot1);
% Author:
%     <Your name>
%     UU
%     Fall 2021
%
```

2. Develop a function,*CS4640_label_not_straight*, described by the header below. Compute $\frac{\partial f}{\partial x}$ and $\frac{\partial f}{\partial y}$ generated by the Matlab gradient function and use these to compute the magnitude of the gradient and the orientation at each pixel. Use the orientation on the edge in the upper label area to determine if the labe is straight. Show mag and ori images and discuss issues with this approach.

```
function b = CS4640_label_not_straight(im)
% CS4640_label_not_straight - bottle label is not straight on middle bott
% On input:
%     im (MxNx3 array): input RGB image
% On output:
%     b (Boolean): 1 if label is not straight; else 0
% Call:
%     b = CS4640_label_not_straight(bot1);
% Author:
%     <Your name>
%     UU
%     Fall 2021
%
```

3. Develop two functions, *CS4640_underfilled* and *CS4640_overfilled*, descibed below. Threshold the image to make the liquid foreground, and find the liquid level using the Canny edge detector. Show examples of these edges on bottle images.

```
function b = CS4640_underfilled(im)
% CS4640_label_underfilled - middle bottle is underfilled
% On input:
%     im (MxNx3 array): input RGB image
% On output:
```

```
%     b (Boolean): 1 if label is underfilled; else 0
% Call:
%     b = CS4640_label_underfilled(bot1);
% Author:
%     <Your name>
%     UU
%     Fall 2021
%


function b = CS4640_overfilled(im)
% CS4640_label_overfilled - middle bottle is overfilled
% On input:
%     im (MxNx3 array): input RGB image
% On output:
%     b (Boolean): 1 if label is overfilled; else 0
% Call:
%     b = CS4640_label_overfilled(bot1);
% Author:
%     <Your name>
%     UU
%     Fall 2021
%
```

4. Develop a function, *CS4640_cap_boundary*, described below. It should return an image with the boundary of the middle bottle cap (if there is one!). Try some combination of techniques already covered to get the region of the cap isolated as the foreground, and then run the closed boundary zero-crossing option in the Matlab function edge to obtain the cap boundary. Show on a few images a quantitative measure of performance for this function (e.g., how many edge pixels are correct).

```
function imo = CS4640_cap_boundary(im)
% CS4640_cap_boundary - extract edge of cap boundary
% On input:
%     im (MxNx3 array): input RGB image
% On output:
%     imo (MxN array): binary image with 1's at cap boundary
% Call:
%     imo = CS4640_cap_boundary(bot1);
```

```
% Author:
%      <Your name>
%      UU
%      Fall 2021
%
```