Team Number:	apmcm2105448
Problem Chosen:	A

2021 APMCM summary sheet

apmcmthesis LaTeX template is designed by https://www.latexstudio.net for http://www.apmcm.org.

This paper presents a simple low-cost first stage solution approximation aimed at ultimately achieving industry level accuracy which could be written as a hobby book for an 11-year-old child.

Keywords: Image gradient magnitude pixel-pixel map-sum Symmetry

Contents

I. Introduction

The problem is reported to have originated from the following background.

1.1

Currently, a efficient model to extract and measure image features with poorer quaity input images is under heated discussion. Modern efficient means with just an image are replacing traditionally physical measurings. In general, a specific calibration plane is required.

1.2

Accurate image edges are crucial rules in feature engineering which is a gap usally steep between background and target features with noises reduced. Both important to human instincts and image processing. Inside the blackbox is unknown yet vital to all aspects.

1.3

Earliest efforts approximate object contours in images with a degree of polygons. Although solvers like canny derive edge information with the help of magnitude gaps present in image grayscale, it is designed to return discrete points not a continuous object. Also sub-pixel calculations are reported to be under heated discussion.

II. The Description of the Problem

2.1 How do we detect, denoise and segment edge curves in sub-pixel level with sufficient robustness given a grayscale image to extract more value or reduce cost in applications?

2.2 How do we calibrate the product images efficiently with given dot plane views and provide a more accurate actual arc-length approximation?

2.3 How could we quickly and accurately generate a complex target curve with limited fundamental local fitting patterns to rapidly deploy to applications?

III. Models

3.1 Basic Model

3.1.1 *Terms, Definitions and Symbols*

The signs and definitions are mostly generated from an intersection of linear algebra, multivariate calculus and classic computer vision.

3.1.2 *Assumptions*

- The working principles of the presented pixel-wise edge extraction model could be transfered and extended to sub-pixel image analysis.
- Occam's Razor theory
- We inherited classic computer vision tools to encapsulate a more advanced model, which is similar to modern advanced software engineering.
- We use symmetry properties of ideal curves to reduce computational cost within an accuracy tolerance.

3.1.3 *The Foundation of Model*

1) The utility function (matlab):

- imsharpen()
- histeq()
- imgaussfilt()
- imcontrast
- edge(img, 'Canny')
- imgradient(img, 'central')
- regionprops()
- plot toolbox

3.1.4 *Solution and Result*

1) Results: Q1:

Table 1. Pic1_1 Edge Contour Data Output Format

Total Edge Contours Count		1
Total Edge Contours Length		802.675749999999
Edge Contour 1	Length	1
	PointCount	802.675749999999
Edge Contour 2	Length	
	PointCount	
…	…	…

Table 2. Pic1_2 Edge Contour Data Output Format

Total Edge Contours Count		6
Total Edge Contours Length		954.5105
Edge Contour 1	Length	
	PointCount	
Edge Contour 2	Length	
	PointCount	
…	…	…

Table 3. Pic1_3 Edge Contour Data Output Format

Total Edge Contours Count	2 correct + 138 noise
Total Edge Contours Length	4.245877500000000e+02

Q2:

Table 4. Edge Contour Length Output Format (mm)

Contour ID	Length(mm)
Total Edge Contours	304.5
Edge Contour 1	156.6
Edge Contour 2	26.55
Edge Contour 3	22.76
Edge Contour 4	27
Edge Contour 5	25.23
Edge Contour 6	46.32

Q3: poor time-management of the author means no implementation with only demo data

Table 7. Contour 1 Segmentation Data Output Format

Edge Contour 1							
NO	TYPE	PARAM					
S1							
S2							
S3							
S4							
S5							
...							

Table 8. Contour 2 Segmentation Data Output Format

Edge Contour 2							
NO	TYPE	PARAM					
S1							
S2							
S3							
S4							
S5							
...							

Strength: Cheap

Weakness: Currently incomplete(not sub-pixel), inaccurate(for a1.bmp noises and not much image calibration with results derived by pixel counting and summing up) and relies on hard-coding to segment contour objects

IV. Conclusions

4.1 Methods used in our models

- Image gradient magnitude
- Pixel-pixel map-sum
- Symmetry

V. Future Work

5.1 Non-linear curve fit optimization with an elegant target function

VI. References

[1] Wang Guojun, A sub-pixel circle detection algorithm combined with improved RHT and fitting[J], Multimedia Tools and Applications, 2020, 79, 29825–29843.

VII. Appendix

Paper written and generated via LaTeX, free distribution. Graph generated and calculation using MATLAB R2020b.

Listing 1: The matlab Source code for problem A question 1

```
clear % mem clear
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
img = imread('a1.bmp');
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% g = imread('a2.bmp');
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% read and preprocess for a3 denoised only
% img = imread('a3m.bmp');
% img = rgb2gray(img);
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% img pre-processing for a1, a3
s = imsharpen(img,'Radius',2,'Amount',1);
he = histeq(s);
g = imgaussfilt(he,1.4);
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%GUI tool used to find the best value of eliminating outliners to
    generate a3 denoised image
% imshow(g)
% imcontrast
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%used only for a3 final output
% b = edge(img, 'Canny');
% imshow(img)
% hold on
% visboundaries(b)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% % close operation -> bugs -> disabled
% se = strel('disk',2);
% % g = imbothat(g,se); % bottom hat operation -> bug -> disabled
% imgC = imclose(g,se);
% imshow(imgC);
```

```
% %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
[Gmag, Gdir] = imgradient(g, 'central');
% %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% % edge outline storage matrix for a1, a2
b = false(size(g));
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% a3 only
% bound = false(size(b));
% %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% edge threshold extract for a1
for i=1:972
    for j=1:1276
        if Gmag(i,j)>30
            b(i,j) = 1;
        end
    end
end
% %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% % edge threshold extract for a2
% for i=1:972
%     for j=1:1276
%         if Gmag(i,j)>20 && Gmag(i,j)<80
%             b(i,j) = 1;
%         end
%     end
% end
% % %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% % edge threshold extract for denoised a3 image: a3m
% for i=1:667
%     for j=1:1000
%         if b(i,j)>=6
%             bound(i,j) = 1;
%         end
%     end
% end
% %
    %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

```
% for a1, a2 only
imshow(g)
hold on
visboundaries(b)
% %
    %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% arc length approximate
perim = regionprops(b,'Perimeter');
perim = perim.Perimeter;
% by definition we /=2 then we approximate the arc length by /= 2 again
    since my model would draw the edge about twice as it should be.
perim = perim/4;
% %
    %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% point cnt and
cnt = 0;
s = size(b);
r = s(1);
c = s(2);
for i=1:r
    for j=1:c
        if b(i,j) == 1
            cnt = cnt + 1;
        end
    end
end
cnt = cnt/2;
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% extract x, y from boundary bool matrix
out = zeros(7000,2);
s = size(b);
r = s(1);
c = s(2);
k = 1;
for i=1:r
    for j=1:c
        if b(i,j) == 1
```

```
            out(k,1) = i; out(k,2) = j;
            k = k+1;
        end
    end
end
```

Listing 2: The matlab source code for problem A question 2

```
clear % mem clear
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
img = imread('b4.bmp');
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% g = imread('a2.bmp');
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% read and preprocess for a3 denoised only
% img = imread('a3m.bmp');
% img = rgb2gray(img);
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% img pre-processing for b
s = imsharpen(img,'Radius',2,'Amount',1);
he = histeq(s);
g = imgaussfilt(he,1.4);
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%GUI tool to find the best value of eliminating outliners to generate
    a3 denoised image
% imshow(g)
% imcontrast
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%used only for a3 final output
% b = edge(img, 'Canny');
% imshow(img)
% hold on
% visboundaries(b)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% % close operation -> bugs -> disabled
% se = strel('disk',2);
% % g = imbothat(g,se); % bottom hat operation -> bug -> disabled
% imgC = imclose(g,se);
```

```
% imshow(imgC);
% %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
[Gmag, Gdir] = imgradient(g, 'central');
% %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% % edge outline storage matrix
b = false(size(g));
% %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% edge threshold extract for b
for i=1:972
    for j=1:1276
        if Gmag(i,j)>30
            b(i,j) = 1;
        end
    end
end
% %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% % edge threshold extract for a2
% for i=1:972
%     for j=1:1276
%         if Gmag(i,j)>20 && Gmag(i,j)<80
%             b(i,j) = 1;
%         end
%     end
% end
% % %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% % edge threshold extract for denoised a3 image
% % for i=1:667
% %     for j=1:1000
% %         if Gmag(i,j)>40 && Gmag(i,j)<120
% %             b(i,j) = 1;
% %         end
% %     end
% % end
% %
    %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
imshow(g)
hold on
```

```
visboundaries(b)
% %
    %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% arc length approximate
perim = regionprops(b,'Perimeter');
perim = perim.Perimeter;
% by definition we /=2 then we approximate the arc length by /= 2 again
    since my model would draw the edge about twice as it should be.
perim = perim/4;
% %
    %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% point cnt per contour
cnt = 0;
s = size(b);
r = s(1);
c = s(2);
for i=1:r
    for j=1:c
        if b(i,j) == 1 && j>=766 && j<=1075 && i>=403 && i<=612
            cnt = cnt + 1;
        end
    end
end
cnt = cnt/2;
```

Listing 3: The matlab source code for problem A question 2

```
clear
c1 = readtable('c1.xls');
c1 = table2array(c1);
c2 = readtable('c2.xls');
c2 = table2array(c2);
x = c1(:,1);
y = c1(:,2);
[Gmag, Gdir] = imgradient(c1);
figure
plot(c1(:,1), c1(:,2), c2(:,1), c2(:,2))
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

```
% lsqcurvefit() %curve fit prototype
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```