Dela Cruz, Mary Nathalie G
2015-09114

The objective of this activity is to obtain the area of 2d objects through image processing via Green's Theorem. Green's Theorem relates a double integral over a region R bounded by a curve C to a line integral around a curve C with the following equation:

$$\iint \left(\frac{\partial F_2}{\partial x} - \frac{\partial F_1}{\partial y}\right) dx\, dy = \oint (F_1\, dx + F_2\, dy) \qquad [1]$$

where F_1 and F_2 are continuous functions that have continuous partial derivatives everywhere in region R. This is applied to area measurement by turning the double integral that defines the area into a line integral. This was achieved by letting $F_1 = 0,\ F_2 = x$ and $F_1 = -y,\ F_2 = 0$. By averaging the resulting equations, we can get the area A of the region R, which is given by

$$A = \frac{1}{2}\oint (x\, dy - y\, dx) \qquad [2]$$

In discrete form, this equation becomes

$$A = \frac{1}{2}\sum_{i=1}^{N}[x_i y_{i+1} - y_i x_{i+1}] \qquad [3]$$

where N is the number of pixels in boundary curve C. [1]

Below was the code implemented to measure the area in pixels squared via Green's Theorem. Line 1 reads the image from a file as a matrix of color values. In this activity, we used BMP images of simple shapes made using Paint and land areas from Google Maps. The edges of an input image were then detected in Line 2. There are different edge detection methods available in Matlab. We analyzed some of these methods in this activity to determine which among them is the most accurate. The pixel coordinates of the edges were extracted in Line 3. In this line, the find() function returns the indices of the nonzero values in the matrix. Edges were extracted because they represent the closed curve C that bounds the region we want to measure. Before the edge pixel coordinates were used to find the area, they were first sorted in order of increasing angle. To do this, the centroid of the shape was obtained using lines 4 and 5. In line 4, the regionprops() function returns the properties of connected regions in the image. The properties include the pixel coordinates of the centroid of the region, which were then stored in an array by line 5. The pixel coordinates of the centroid were then subtracted to the edge pixel coordinates in line 6. The resulting pixel coordinates were then converted from cartesian to polar in line 7 to get their corresponding angles. They were then sorted according to angle in line 9. In line 11, Green's Theorem was applied with the sorted pixel coordinates to get the computed area A $_{\text{computed}}$ of the image region. Here, we used equation 3. For comparison, the number of white pixels was counted in line 12. This serves as the analytical area A $_{\text{analytical}}$ of the image region. [2]

```
1 shape = imread('circle1.bmp'); %reads image
2 edge = edge(shape,'Roberts'); %gets the edge Booleans of the image
3 [Y,X] = find(edge); %gets the coordinates of the edges
4 stats = regionprops (edge, 'centroid'); %gets centroid for connected components in image
5 centroids = cat(1, stats.Centroid); %stores x and y coordinates of the centroid
6 x = X-centroids(:,1); y = Y-centroids(:,2);
7 [angle,radius] = cart2pol(x,y); %transforms cartesian coordinates to polar coordinates
8 pixel = [angle,radius,x,y];
9 sorted = sortrows(pixel,1); %sorts according to the angle
10 xs = sorted(:,3); ys = sorted(:,4); %cartesian coordinates arranged by angle
11 area = abs(sum(xs.*ys([2:end,1])-ys.*xs([2:end,1])))/2; %applies Green's Theorem
12 area_theo = nnz(shape==1); %computes the analytical area
13 error = (area_theo-area)/area_theo; %computes the percent error
```

Figure 1 shows the synthetic images of different shapes and their edges that were detected via Canny method, Sobel method, Prewitt method and Roberts method. Canny method detects edges by getting the local maxima of the gradient of the shape, while the last three methods use different approximations to the derivative in order to find the edges. The differences of the detection methods are not evident in Figure 1. Hence, for every detection method, we obtained the area computed using Green's theorem A $_{\text{computed}}$ and compared each of them to the analytical area A $_{\text{analytical}}$, which is the number of white pixels. The areas are listed in Table 1. As shown in the table, all methods are viable because the % deviation between the computed area and the analytical area of each method is less than 1. Among the 4 detection methods analyzed, we used the Roberts method because it yields no percent error. [2]

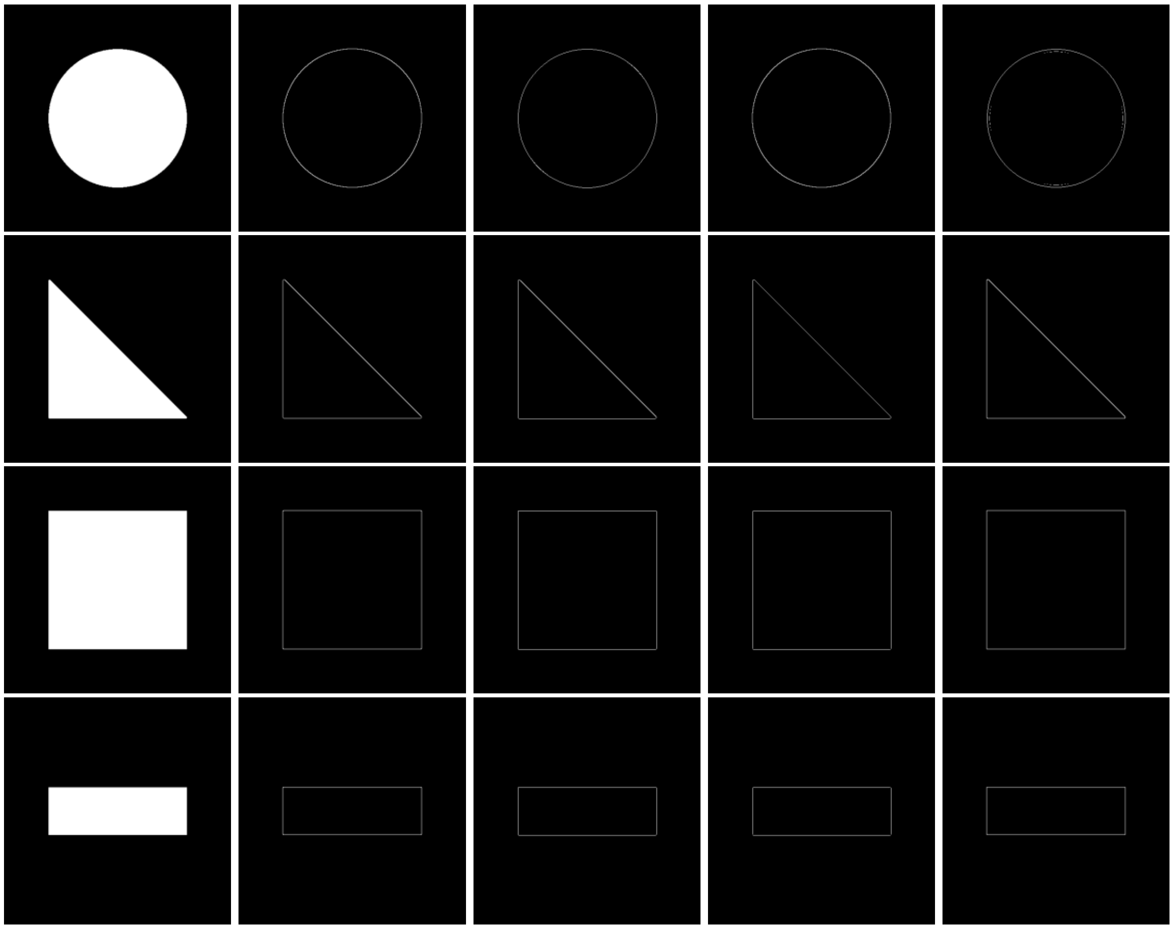


Figure 1: Black and white synthetic images (row 1) of a circle (column 1), a triangle (column 2), square (column 3), and rectangle (column 4) and their edges detected using Canny method (row 2), Sobel method (row 3), Prewitt method (row 4) and Roberts method (row 5).

Table 1: Area computed using Green's theorem A $_{\text{computed}}$, analytical area A $_{\text{analytical}}$, which is the number of white pixels, and the % error for the different shapes (circle, triangle, square, and rectangle) and different edge detection methods (Canny, Sobel, Prewitt and Roberts).

		Canny		Sobel		Prewitt		Roberts	
Shape	A $_{\text{analytical}}$ (pix^2)	A $_{\text{computed}}$ (pix^2)	% error	A $_{\text{computed}}$ (pix^2)	% error	A $_{\text{computed}}$ (pix^2)	% error	A $_{\text{computed}}$ (pix^2)	% error
Circle	72969	72969	0	72964	0.001713	73000	0.010619	72969	0
Triangle	47571	47567	0.002102	47566	0.002628	47716	0.076086	47571	0
Square	93025	93021	0.001075	93022	0.000806	93021	0.001075	93025	0
Rectangle	32035	32021	0.003123	32022	0.002342	32020	0.003904	32035	0

Figure 2 shows that the synthetic images of a diamond, hexagon and pentagon, and their edges that were detected via Roberts method. As shown in Table 1, the percent deviation between the area computed using Green's theorem A $_{\text{computed}}$ and the analytical area A $_{\text{analytical}}$ is zero for all shapes when using the Roberts edge detection method. Hence, in the next part of the activity, we again used Roberts method in detecting the edges of land areas from Google Maps.

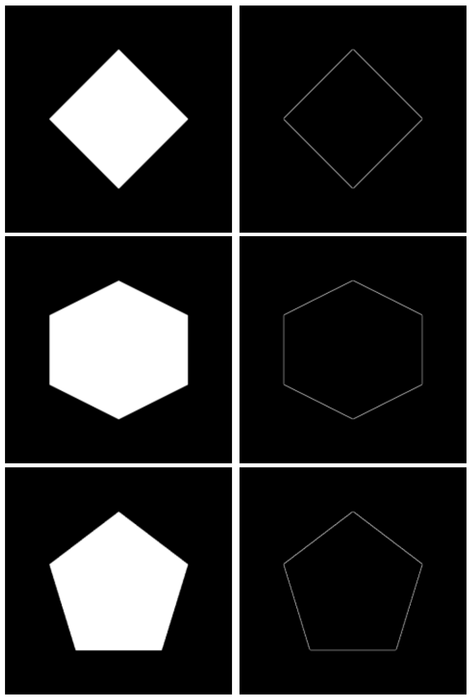


Figure 2: Black and white synthetic images (row 1) of a diamond (column 1), a hexagon (column 2), and a pentagon (column 3), and their edges detected using Roberts method (row 2).

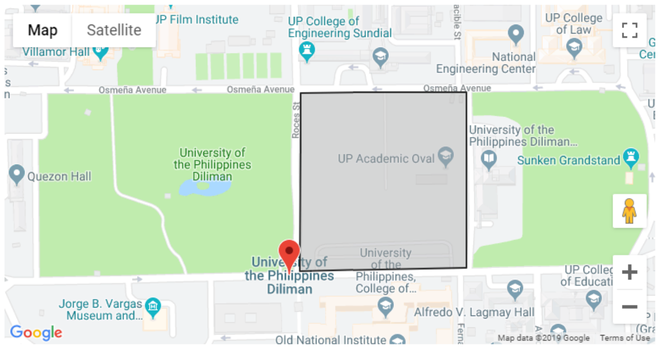
Table 2: Area computed using Green's theorem A $_{computed}$, analytical area A $_{analytical}$, which is the number of white pixels, and the % error for a diamond, hexagon, and pentagon.

Shape	A $_{analytical}$ (pix^2)	A $_{computed}$ (pix^2)	Percent Error
Diamond	47121	47121	0
Hexagon	69921	69921	0
Pentagon	64695	64695	0

To measure the land area from Google Maps in meters squared, lines 1-13 were implemented along with lines 14-17. Lines 14-17 converts the image area in pixels squared to the land area in meters squared. Line 14 defines the length and width in meters of the area encompassing the image. They were measured using reference 3. Line 15 gives the x and y dimension size of the image. They were then divided to the land area dimensions in line 16 to get the conversion factor. In line 17, the image area in pixels squared was then converted to the land area in meters squared using the conversion factor.

```
14 xmax = 890; ymax = 470; %meters
15 [ny,nx] = size(shape); %pixels
16 convx = xmax/nx; convy = ymax/ny; %conversion factor
17 conv_area = area*convx*convy; %converts image area from pixel squared to meters squared
```

The land area of a part of the academic oval and the area of the field in Ateneo Senior Highschool were measured using the algorithm above. Shown in Figure 3 is the measured area of Academic Oval, the delineated version of that area obtained using Paint, and its edges detected using the Roberts method. On the other hand, Figure 4 shows the measured area of Ateneo Senior Highschool field, the delineated version of that area obtained using Paint, and its edges detected using the Roberts method. The area in meters A $_{online}$ was measured online using reference 3. It was compared with the area computed using Green's Theorem A $_{computed}$ in meter squared units. The computed area was also compared to the analytical area A $_{analytical}$ in meter squared units. Table 3 shows these areas and table 4 lists the percent error between the area obtained online A $_{online}$ and the computed area A $_{computed}$, and the percent error between the analytical area A $_{analytical}$ and the computed area A $_{computed}$. As shown from Table 4, the % errors between the computed area and the analytical area for both locations are zero. This means that the algorithm was able to give an accurate area in pixels squared from the delineated version of the location. However, the % error between the computed area in meters squared and the area obtained online is nonzero. This may be due to (1) error in defining the boundaries during delineation, (2) error in the calculated conversion factor, and (3) error in the area obtained online. [3]


(a)

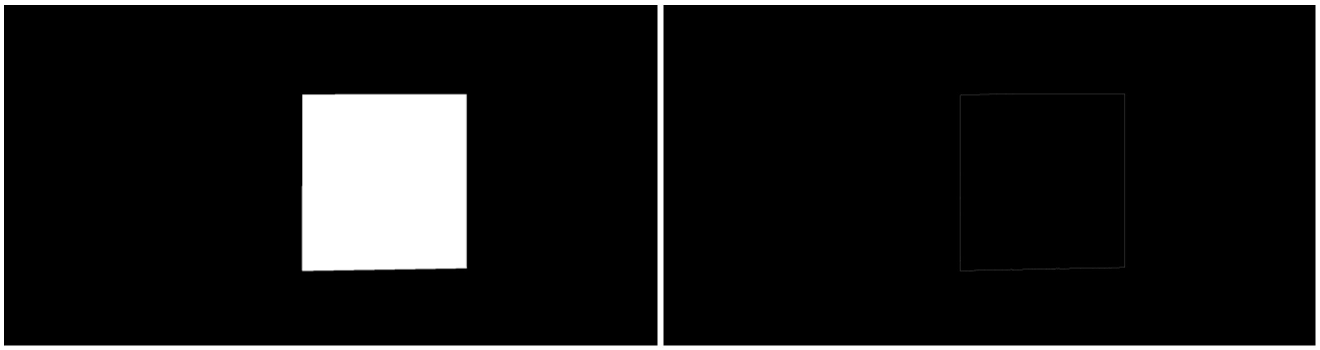
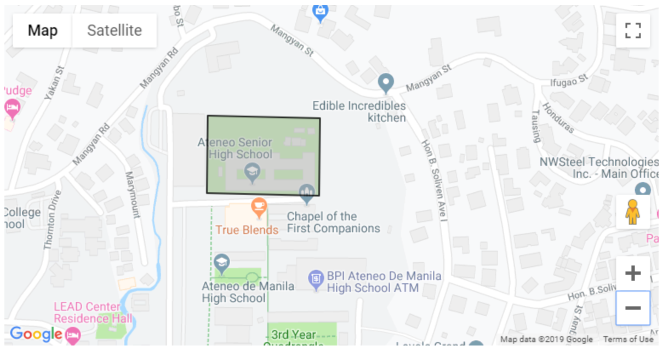

(b) (c)

Figure 3: (a) Measured area of Academic Oval, (b) the delineated version of that area, and (c) its edges detected using the Roberts method.


(a)

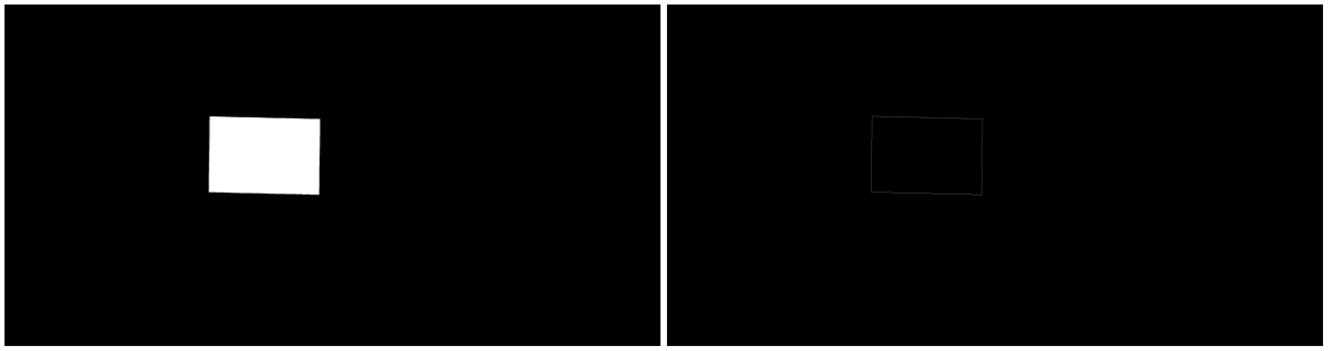

(b) (c)

Figure 4: (a) Measured area of Ateneo Senior Highschool field, (b) the delineated version of that area, and (c) its edges detected using the Roberts method

Table 3: Analytical area $A_{analytical}$ and computed area $A_{computed}$ in pixels squared and in meters squared, and area from online A_{online} in meters squared.

	$A_{analytical}$ (pix^2)	$A_{computed}$ (pix^2)	$A_{analytical}$ (m^2)	$A_{computed}$ (m^2)	A_{online} (m^2)
Academic Oval	1083791	1083792	54280	54280	54698
Ateneo Field	310271	310271	16118	16118	15637

Table 4: % error between computed area $A_{computed}$ and analytical area $A_{analytical}$, and % error between computed area $A_{computed}$ and area obtained online A_{online}.

	% error between $A_{computed}$ (m^2) and $A_{analytical}$ (m^2)	% error between $A_{computed}$ (m^2) and A_{online} (m^2)
Academic Oval	0	0.764196
Ateneo Field	0	2.984241

References

[1] http://tutorial.math.lamar.edu/Classes/CalcIII/GreensTheorem.aspx
[2] https://in.mathworks.com/help/index.html
[3] https://www.mapdevelopers.com/area_finder.php

Self-evaluation: Technical correctness: 5, Quality: 5, Initiative: 2