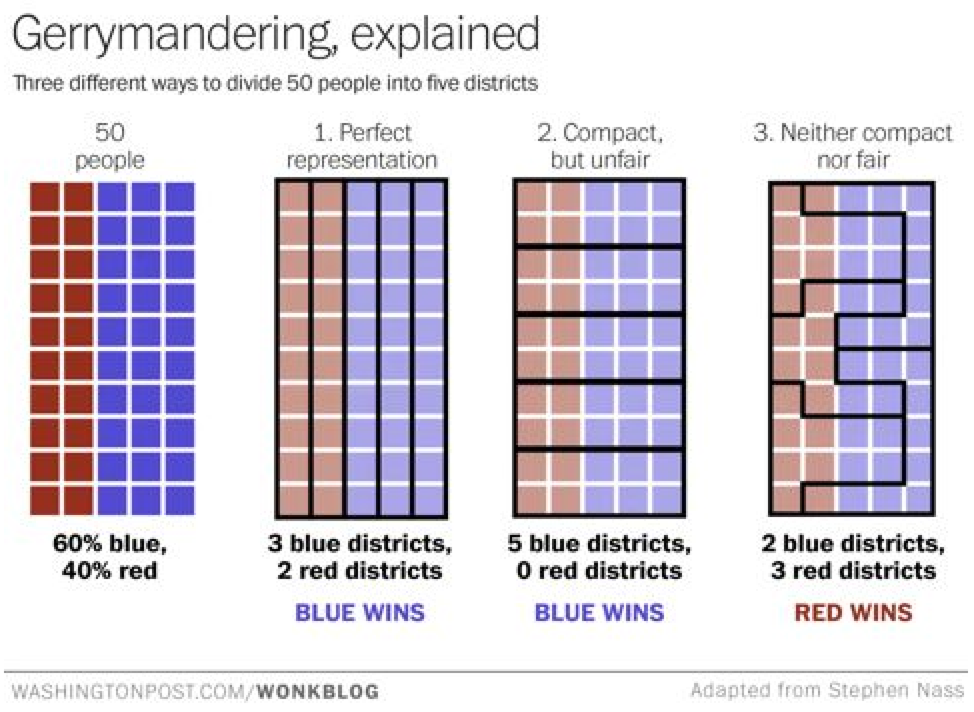
Arjun Rao arrao
AMS 545/CSE 555 (Spring, 2019)
Computational Geometry: Final Project Report - Niceness of a Polygon 5/9/19

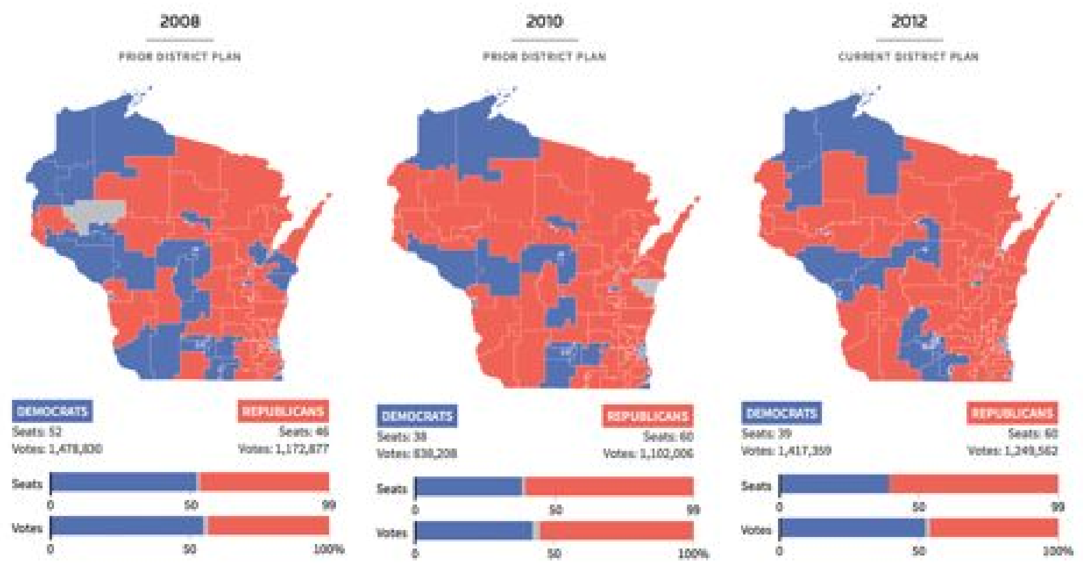
Niceness of a Polygon: Interactive Python Application

How do we measure the niceness of a polygon? First we need to determine what "niceness" means. As stated in the list of potential project descriptions, there are various notions of quantifying how "nice" or how "fat" a simple polygon P is. Niceness is similar to compactness, a concept explored in gerrymandering. Thus, this question has real-life relevance in terms of quantifying how "compact" polygonal election districts are.

Gerrymandering is a political technique in which lawmakers will "manipulate the boundaries of (an electoral constituency) so as to favor one party or class." The concept of how this can be done is explained as follows:


Gerrymandering, explained
Three different ways to divide 50 people into five districts
50 people
1. Perfect representation
2. Compact, but unfair
3. Neither compact nor fair
60% blue, 40% red
3 blue districts, 2 red districts
BLUE WINS
5 blue districts, 0 red districts
BLUE WINS
2 blue districts, 3 red districts
RED WINS
WASHINGTONPOST.COM/WONKBLOG
Adapted from Stephen Nass

Thus, it benefits lawmakers to pack a lot of people of their opposing party into a few districts and spread the rest out across the other districts. We can see the effect of this as shown in Wisconsin over time:



Now that we have some background, we can see that pursuing gerrymandering often causes lawmakers to construct very wonky-looking districts. If assessed for compactness, these districts would be judged as “less compact” or “less nice” than other districts. Thus, being able to assess the compactness of a polygon is a useful metric. A very helpful article that explains these concepts is the following paper by Gary King that describes compactness and various measures of it: https://gking.harvard.edu/files/gking/files/compact.pdf

There are a few measures of compactness that were used - for their intuitiveness, popularity, and visual simplicity - since we will be making an interactive visual app. As presented in the app, I calculated the ratios of the area of the polygon to: the area of the convex hull; the area of a minimum bounding/enclosing circle; and the area of the bounding box

You can run the app with the following command (note that python 3.6 was used):

jupyter nbconvert --to script CompGeoPolygonNicenessApp.ipynb; python CompGeoPolygonNicenessApp.py

Or if you have already have the python file, simply:

python CompGeoPolygonNicenessApp.py

In the app, you can click points to construct your polygon (in clockwise or counterclockwise order). The area of the polygon will be given as feedback to the user. There are also four buttons in the interface, and the first three have a corresponding ratio. They are as follows:

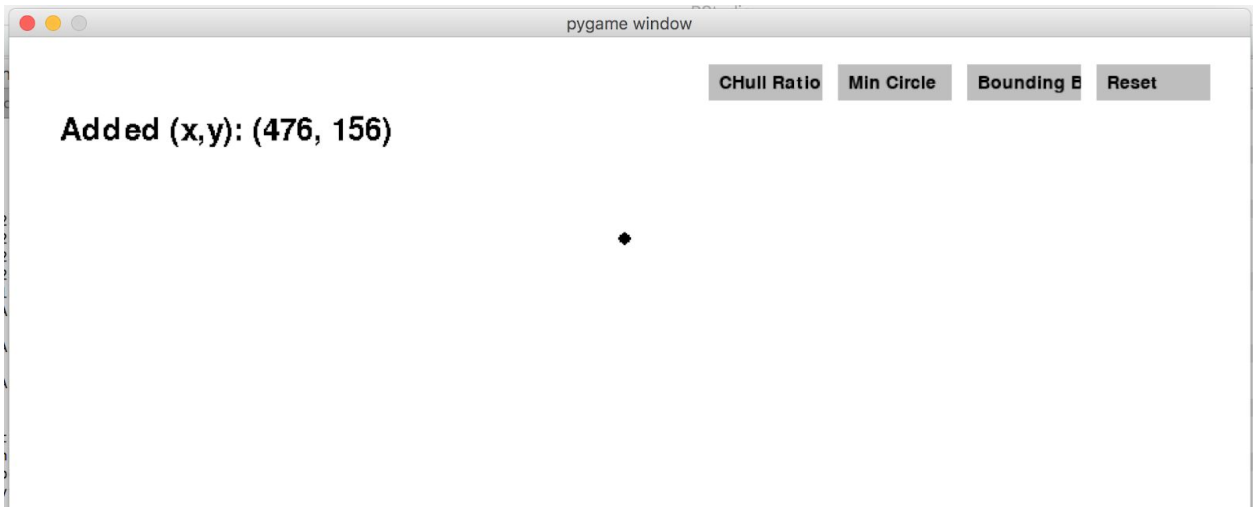
- The first button will calculate and state the convex hull area ratio, and outline the convex hull in blue

- The second button will calculate and state the minimum enclosing circle area ratio, and outline said circle in red
- The third button will calculate and state the bounding box ratio, and outline the box in green
- The last button is a reset button to remove the points and polygon moused in by the user

Note that the "nicest" polygons will be regular n-gons that most closely approximate a circular disk. Their convex hull area ratio will be 1 since they are convex polygons, and the other two ratios will be close to 1 since they are pretty tightly packed into a circular disc and bounding box.

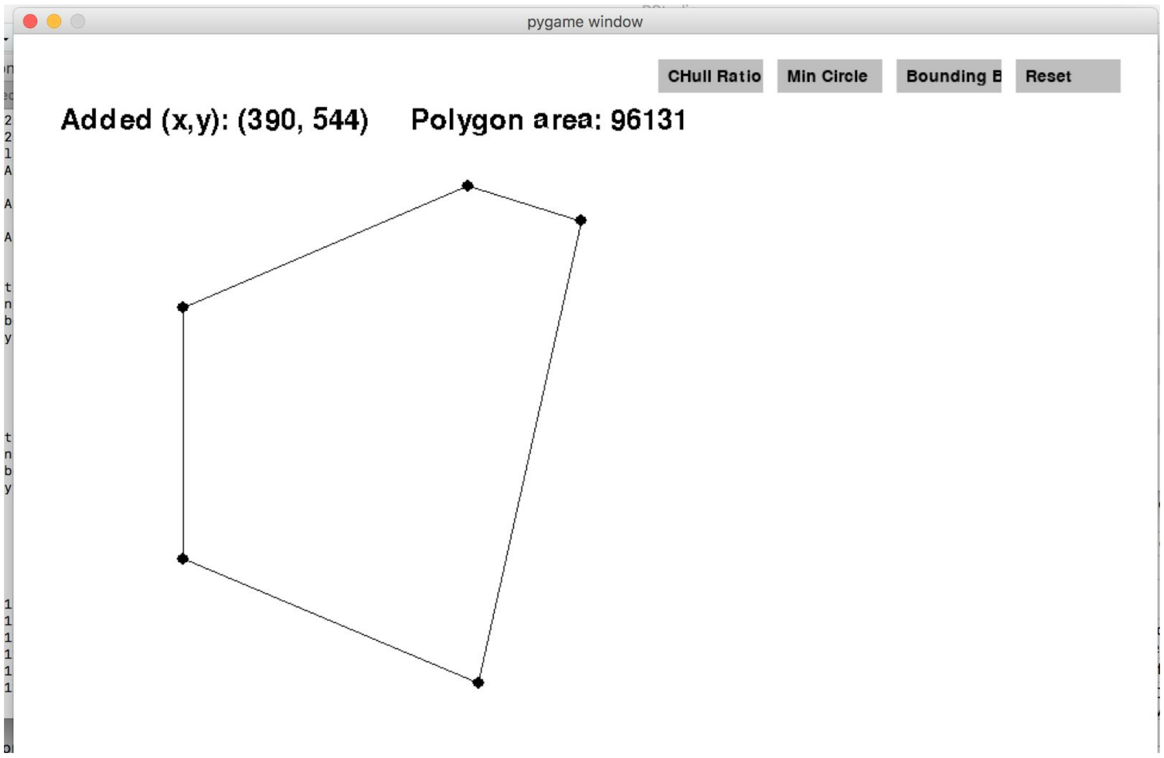
The following is a sequence of screenshots and descriptions of a use of the program:

The app begins with the message: "Click points in cw or ccw order to add." Once you click a point, it will tell you where it was added:



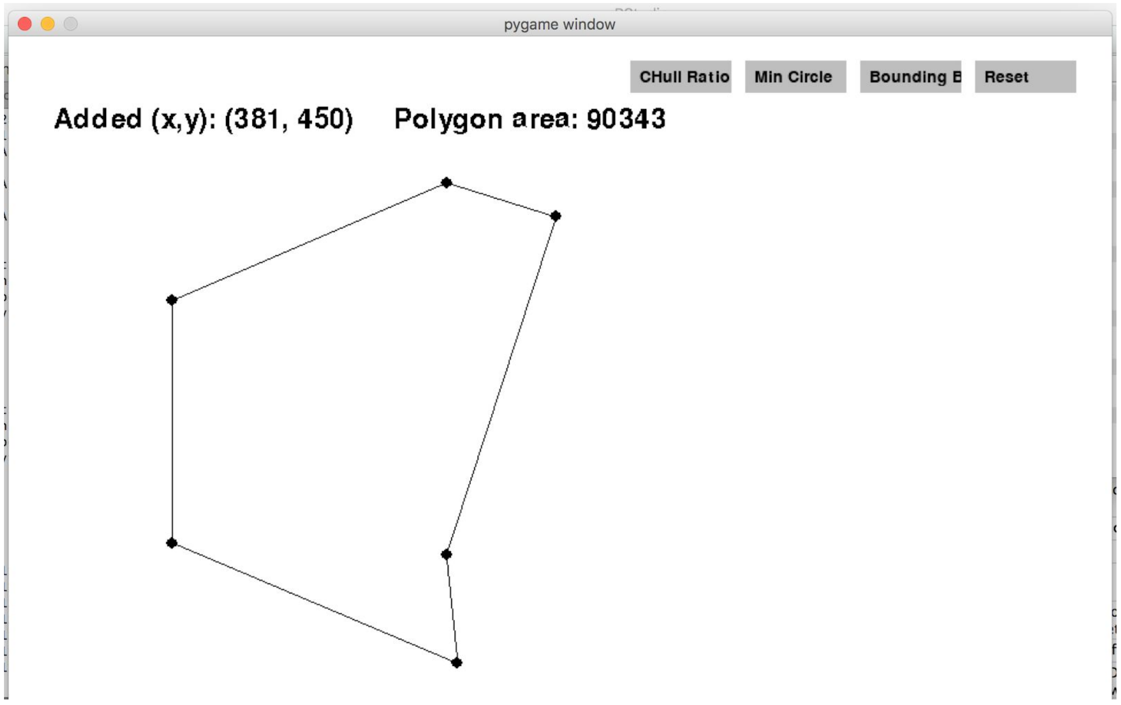
Added (x,y): (476, 156)

As you proceed clicking points (in this case I added them in ccw order), it will build the simple polygon for you. Here is a convex polygon, for which the Convex Hull ratio would of course be 1:



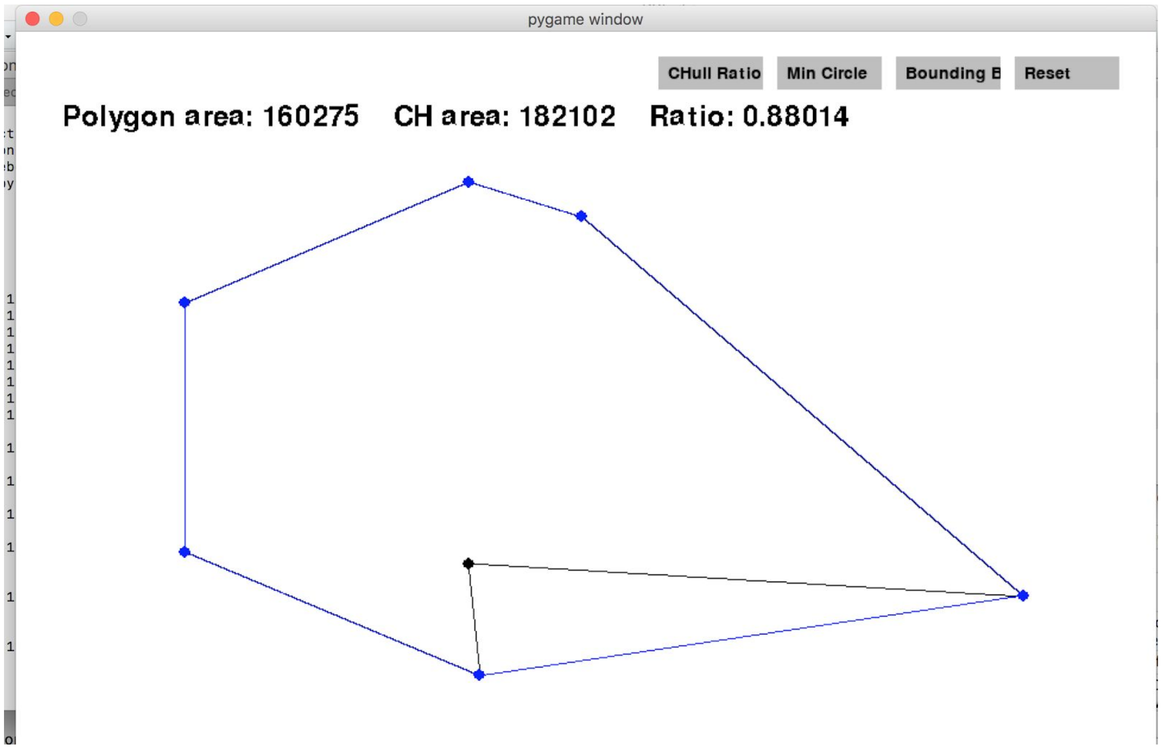
Added (x,y): (390, 544)   Polygon area: 96131

Note that it tells you what the area is. Of course, if you retreat to the interior of the existing polygon with your next point, you will reduce the area of the polygon:



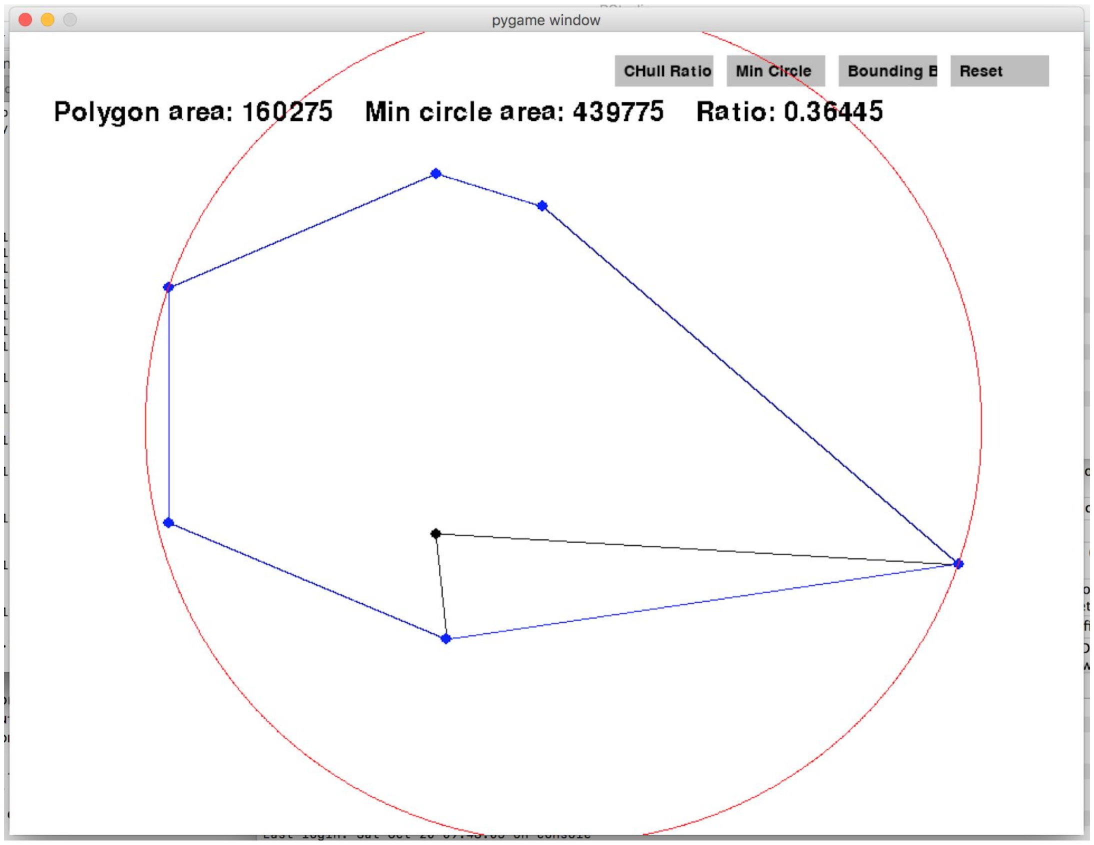
Added (x,y): (381, 450)   Polygon area: 90343

Here we extend the polygon by 1 point and then calculate the CH area ratio, which is close to 1 in this case:



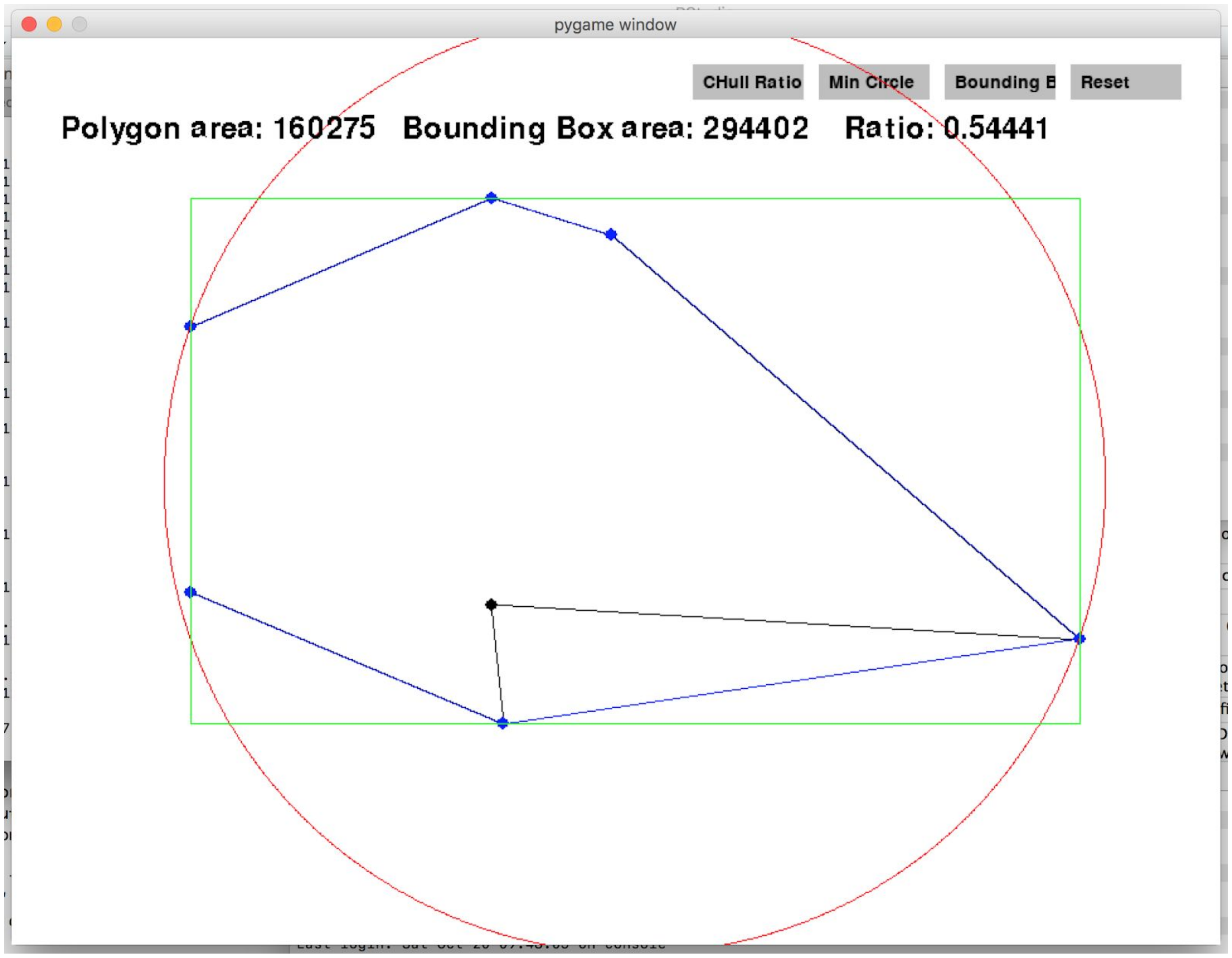
Polygon area: 160275   CH area: 182102   Ratio: 0.88014

Here we calculate the minimum enclosing circle (shown in red) and ratio:



Polygon area: 160275   Min circle area: 439775   Ratio: 0.36445

Here we calculate the bounding box (shown in green) and ratio:

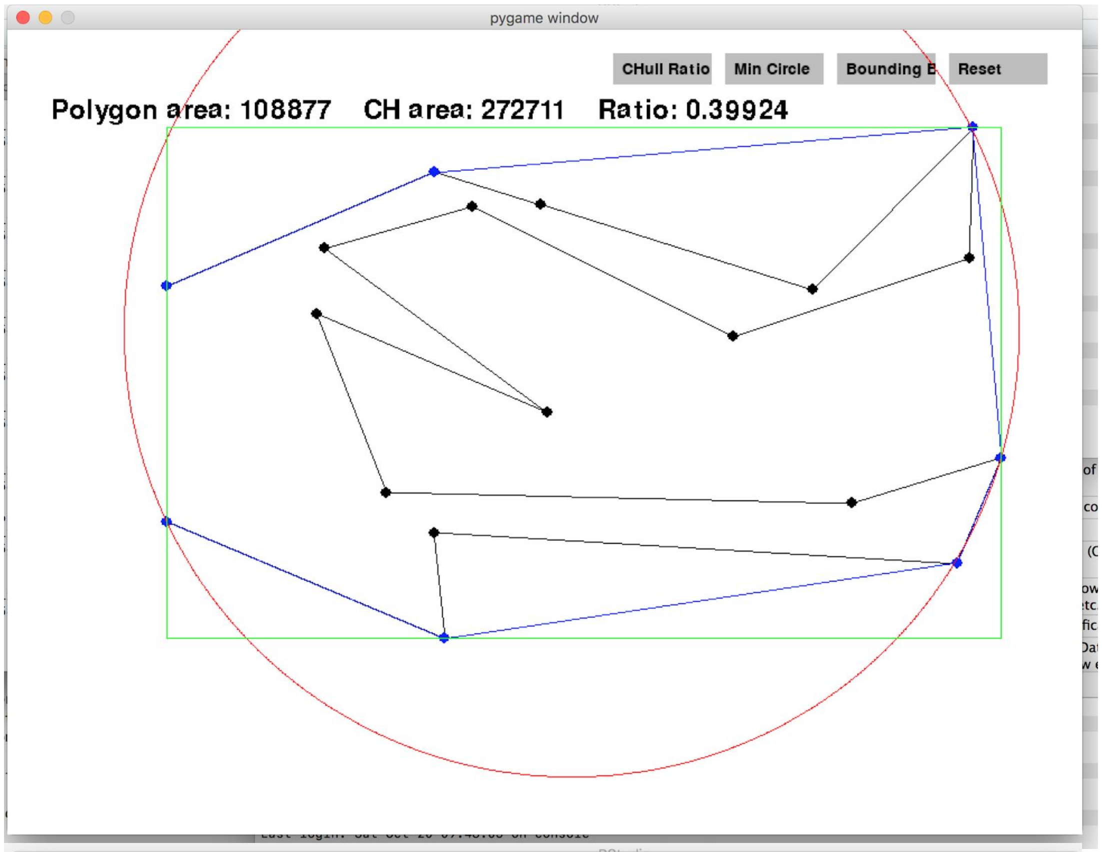


Polygon area: 160275   Bounding Box area: 294402   Ratio: 0.54441

Now, let’s construct a wonkier-looking polygon that we expect would be judged to be “less compact.” Let’s simply add points to our previous polygon.

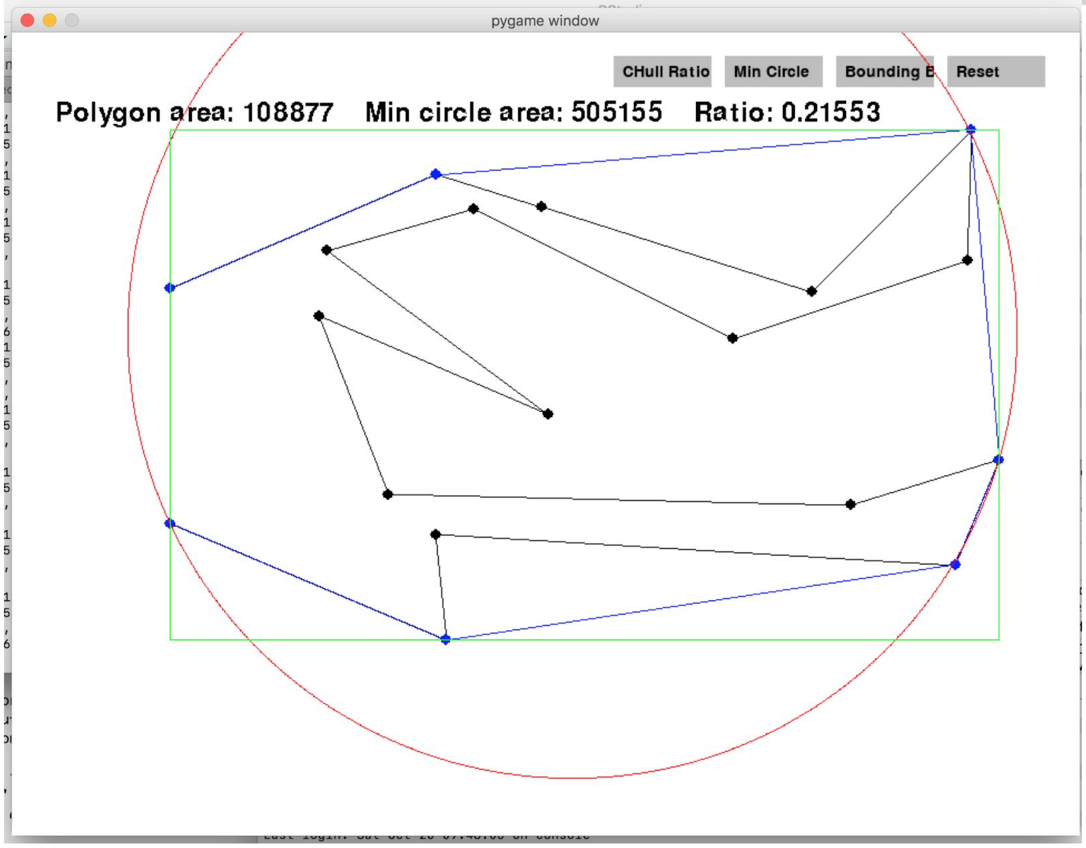
We see that the new convex hull, minimum enclosing circle, and bounding box all seem much larger (proportionally) than the previous polygon's. Here is the new convex hull ratio:

The new convex hull ratio:



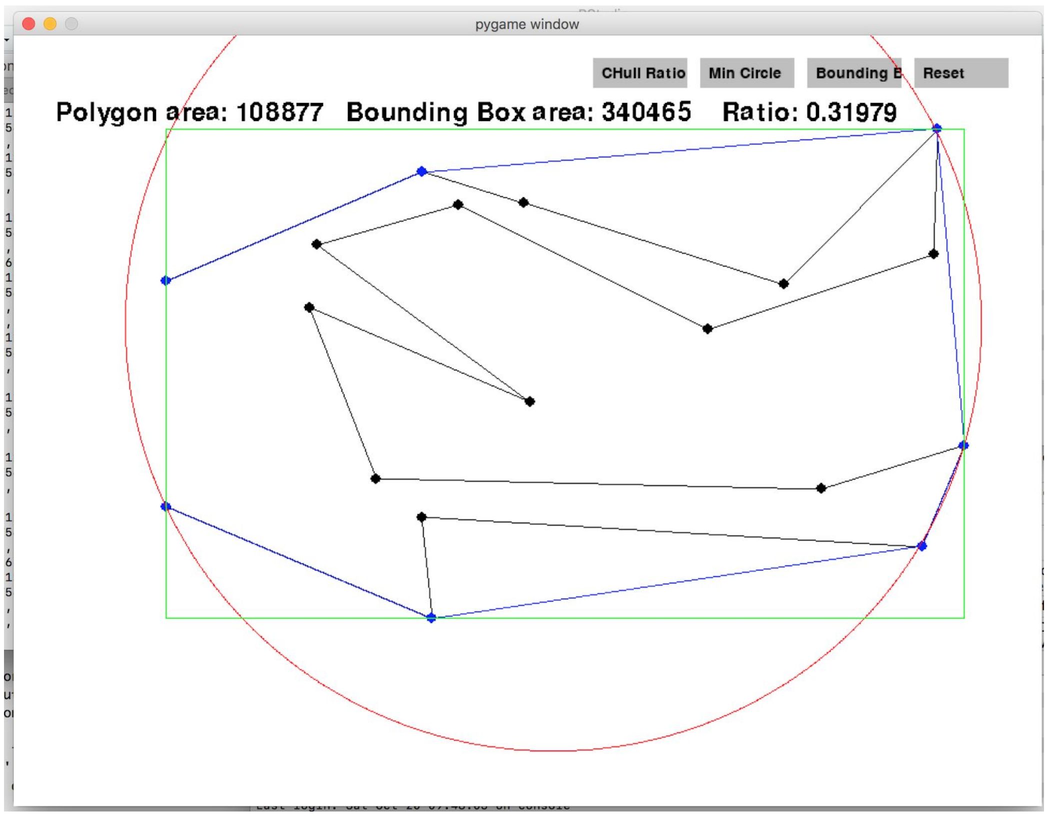
Polygon area: 108877   CH area: 272711   Ratio: 0.39924

The new min circle ratio:



Polygon area: 108877   Min circle area: 505155   Ratio: 0.21553

The new bounding box ratio:



Polygon area: 108877 Bounding Box area: 340465 Ratio: 0.31979

Conclusions:

As we can see, the first/old polygon had rounded ratios of **.88, .36, and .54** (in the order of: the ratio of the area of the polygon to the area of the convex hull, the minimum enclosing circle, and the bounding box).

The new/wonkier polygon had rounded ratios of **.40, .22, and .32** respectively.

As we can see, each measure is lower for the wonkier polygon. This makes sense - we expect lower values as we view this new polygon to be less compact.

Success! The powerpoint and code that I used to demo are attached in a zip file. Credits to certain open source softwares that were used are stated in the opening of the main python / ipynb file.

In the future, there are a few ideas to be pursued. I could make the color-coding more intuitive so it's easier to see the actual polygon. I could be able to upload a file with points in it, which is helpful if you have data on existing polygons. Additional measures of niceness (beyond the three used) could be examined, such as using the area of a circle with the same perimeter as the polygon. Lastly, one could calculate a holistic number for the measures of niceness / compactness that we examined, and weight each metric separately.