Polygan triangulation

Viquez Alpizar Camila and Mata Mora Leonardo

***Abstract*— Results of the Hurtado-Noy method applied to polygon triangulation. Experiments with regular polygons, irregular and convex. Order of the algorithm.**
Emblematic people in the area of computation and a little about what polygons are and why is importand the polygonal triangulation

I. INTRODUCCION

Polygon triangulation is the decomposition of a polygonal area into a set of triangles.The polygon triangulation is an important aspect of computational geometry. In this paper is present the quality of Catalan number use un Hurtado-Noy method, related with polygons, and the properties of polygons for triangulation.

A. Computational geometry

It involves the branch of computer science encharge of the study of algorithms witch can be stated in terms of geometry. Some areas of computational geometry are computer graphics, pattern recognition, image processing, robotics, electronic design automation.

B. Properties for the triangulations of a polygon

- The triangles can not be intersected.
- The union of the triangles is equal to the original polygon.
- The vertex of the triangles are the vertex from the original polygon.
- Every pair of triangles share only one edge or one vertex.

Every polygon always admits at least one triangulation.
A polygon with n vertex has exactly $n-2$ triangles and $n-3$ diagonals.

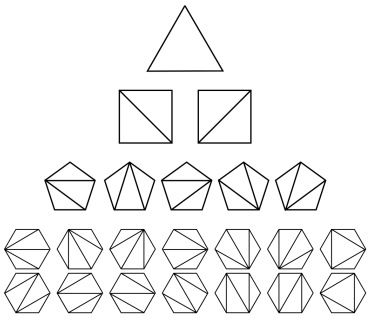

Fig. 1. Polygon triangulation

C. Catalan number

This number is define as $C_n = \frac{(2n)!}{(n+1)!n!}$ It indicates the total number of possible triangulations of a convex polygon, n as the number of triangles, it means $n-2$.
It was proved by Leonhard Euler, he was a Swiss mathematician, physicist and astronomer.

Fig. 2. Leonhard Euler $1707-1783$

II. METHODOLOGY

The programming language for the implementation is Python, version 3.6.2 . Python is an object-oriented and interpreted programming language. The main advantage of Python is its simple syntax and its general simplicity.

The implementation of this algorithm is based on the polygon triangulation properties and the Catalan number.
Given a polygon of n vertex you are able to know the number of diagonals, triangles, and possible combinations of triangles on that specific polygon.

A. Models used for implementation

The Hurtado-Noy algorithm is based on Catalan number. It take a vertex, which is the initial vertex, and visit the other vertex that are not adjacent.
This method implements the polygon properties for diagonals and triangles. So the stop condition for the loop is given by the number of diagonals that a polygon can have, $n-3$, where n is the number of vertex.

B. Data set

$1, 2, 3, ..., nn \in Naturalnumbers.$
Convex polygons of n vertex.

C. Experiments

1) Increase of vertices: This experiment consists of calculating the diagonal triangles and possible combinations of triangles in polygons with different number of vertex. These begin in three, which is the first valid polygon and then the others increase to 10, 40, 60 and 100.

2) Greater number of vertices: This experiment consists of calculating the number of vertex in which the algorithm stops being efficient and its time is not feasible. If you enter a polygon with 19000 vertex the algorithm takes 6 seconds, and if you try 88888 vertex you don't know how long it will take, you might have an answer but in an irrational time.

3) Diagonal distance in irregular polygons: In this experiment we ran the algorithm only with irregular polygon, the ones that have different distance between the edges. This experiment starts form one vertex, and tells you the different distance that have a edge inside the polygon. For irregular polygons the distance between edges can be any distance, so for this experiment we limit the distance in a range from1 to 20, so the distance for the edges is a random on that range.The regular polygons have the same distance between all their vertices, so it does not make sense to analyze them in this experiment. So for this experiment we are going to Analise what happen in five different polygons

D. Results

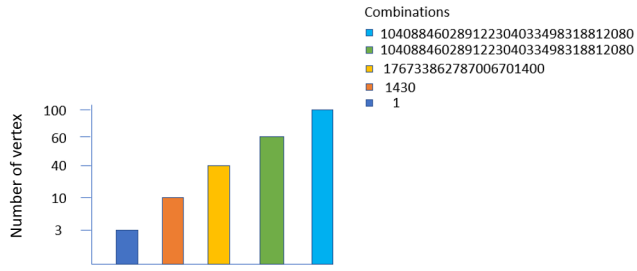

Fig. 3. Increase of vertices

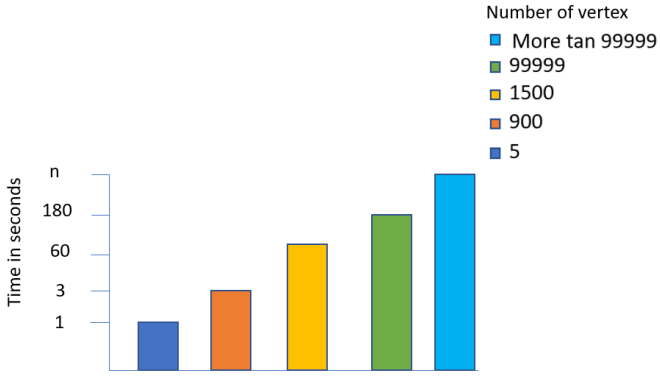

Fig. 4. Diagonal distance in irregular polygons

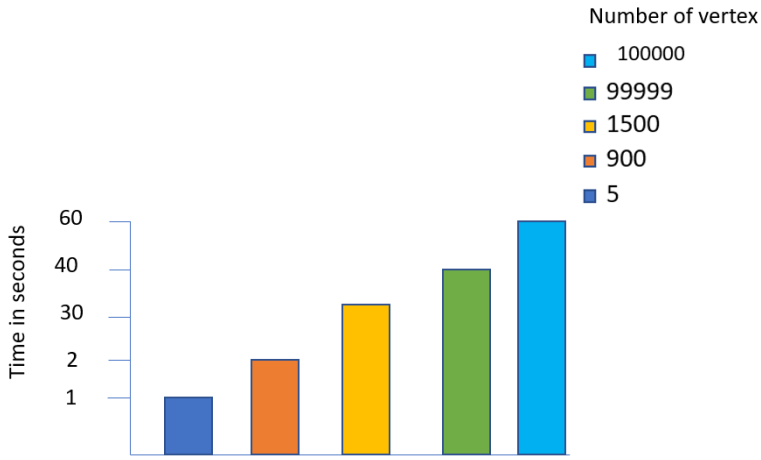

Fig. 5. Greater number of vertices

E. Discussion

On the first experiment we can see that not only the number of vertex increased but also the combination. More vertex in a polygon implies more possibilities.

For the second experiment we run the algorithm until it did not answer. By the polygon theorem we know that if a polygon is regular, it can always be triangulated, so the algorithm is able to responds to a polygon of n vertex, however it does not assure that it is in a feasible time. So in this experiment you can appreciate the pentagon that last one second and the polygon with 1000000 verteces last one minute. The reason why the algorithm takes so much time for polygons with more vertex is because of the Catalan number, this number requires the calculus of the factorial for the number of vertex and two more factorials for the formula. So it is not possible to have the factorial of this number in a short time. from the empirical point of view it also depends of the processor and RAM memory in witch the algorithm is running.

For the third experiment we use only irregular polygons, and we run the same examples that were used on the second example. But the time that implicates the measurement of the distances of a diagonal between two vertex crate a difference, if you compare both experiments you can see how the time increase on irregular polygons.

F. Conclusion

As conclusion is important to see that the practical time is not necessary as the theoretical time, even do on this experiments the practical time is closely relatedly with theoretical time. For example the first experiment, the time of the algorithm depends of the number of vertex. More vertex require more comparison in to a loop, so it affects the barometer variable.

For the second experiment is similar to fort one,but there comes a moment that can not be executed, but this moment is define by the number of vertex, it will happen with really huge polygons.

For the last experiment the number of vertex is the one that defines the barometer of the function, here we can saw a difference between theoretical time and empirical time. Because the distance for the diagonal is just a constant but it doesn't affect the order of the algorithm, the distance has no implications in the loop.

G. Big O and algorithm analysis

1) Principal function: This function ask for the size of the convex polygon and make a call to poliganTriangulation and numCombinations at the end it shows the results. The order of this function is given by the functions that is calling so the big O is define by $O(n)$.

```
def principal():
    edges = int(input("Introduce the number of size for your regular polygon: "))
    firstVertex = int(input("Introduce the number of vertex that you want first: "))
c   numTriangles = edges-2
    numDiagonal = edges-3
    poligonTriangulation(numTriangles, numDiagonal, firstVertex)  -> n
    numCombinationTriangles = catalanNumber(numTriangles)         -> n

    print("Polygon size: ",edges)
    print("Number of posible triangles in the polygon: ", numTriangles)
    print("Number of posible diagonals in the polygon: ", numDiagonal)
    print("Number of posible diferent combinations
          "of triangles in the polygon: ", numCombinationTriangles)
```

Fig. 6. Principal function

2) Polygon triangulation: The barometer of this function is given by the number of diagonals that a convex polygon can have. The order of this function is $O(n)$

```
def poligonTriangulation(numTriangles, numDiagonal, firstVertex):
c   diagonalList = []
    for i in range(numDiagonal):                                  -> n
        c  drawDiagonalIrregular(firstVertex, diagonalList,i+3)
           drawDiagonal(firstVertex, diagonalList,i+3)
```

Fig. 7. Polygon Triangulation

3) Draw Diagonal and Draw Diagonal for Irregular polygon: Booth have an array with the pair of vertex that form a diagonal. The difference is that the $drawDiagoalIrregular()$ implements a random function for the distance . Random function in the background is the same as $x_{n+1} = (ax_n + c)(modm)$ were x_0 is the seed, a is the multiplicative constant c represents the additive constant and m represents the which the rest is calculated. So this represents a constant too. So the contribution of this function to the order is negligible since it is constant and constants are negligible for the order.

```
def drawDiagonal(firstVertex, diagonalList,i):
    diagonalList.append([firstVertex, i])  -> c
    return diagonalList

def drawDiagonalIrregular(firstVertex, diagonalList,i):
c   distance = random.randint(1,20)
    diagonalList.append([firstVertex, i, distance])
    print("Irregular poligon Distance between one vertex to another: " + str(distance))
```

Fig. 8. Draw Diagonal

4) Factorial: This function is required for the calculus of Catalan number. The barometer of this function is given by the number given less one. The big O for this function is $O(n)$.

```
def factorial(number):
    if(number > 0):
        n = number-1
        while(n>0):  ------------------>  n
            number *= n
            n -= 1
    return(number)
```

Fig. 9. Draw Diagonal

5) Catalan Number: This function is the implementation of the original formula for Catalan numbers. This formula requires the factorial of $2 * n, n + 1, n$, the calculus of factorial numbers represents an n in $catalanNumber()$ because $factorial()$ runs a simple loop, so this call affects implicitly the order of the function. So the big O for this function is $O(n)$.

```
def catalanNumber(n):
    a = factorial(2*n)
    b = factorial(n+1)      } n
    c = factorial(n)
    catalanNumber = a // (b*c)  ------>  c
    return(catalanNumber)
```

Fig. 10. catalan number

After this analysis we can say that polygon Triangulation is an algorithm with an order $O(n)$

REFERENCES

[1] Catalan numbers.Brilliant.org. Retrueved 03:05,November,1,2017,from https://brilliant.org/wiki/catalan-numbers/

[2] Predrag S. StanimiroviFaculty of Science and Mathematics. Decomposition of Catalan numbers and convex polygon triangulations University of Ni, Viegradska 33, 18000 Ni, Serbia, from http://www.tandfonline.com/doi/abs/10.1080/00207160.2013.837894