

Name: LE Thu Huong  
Class: ADEO1

## FINAL PROJECT REPORT FOR HEURISTIC

# PROBLEM: Maximization the rectangle inside a polygon

### 1. Ideal explanation:

getBounds() to get the boundary of a polygon: return xmin, xmax, ymin, ymax

pos2rect() to get the point a,b,c,d of rectangle : return a,b,c,d

\*However, function require to rotate the rect to generate all possible solution for SA algr()

\* Main math to apply: please watch this great video (15min) of Khan academy

<https://www.khanacademy.org/math/linear-algebra/matrix-transformations/lin-trans-examples/v/linear-transformation-examples-rotations-in-r2>

distance() return euclidean distance

area() return area of rectangle

verifConstraint() return a sharp ( return true to 4 point, others are return false)

\* install clipper in Linux \$ pip install clipper

\* <https://pypi.org/project/pyclipper/>

initOne() generate a random solution

Other functions are realization of existing algorithms

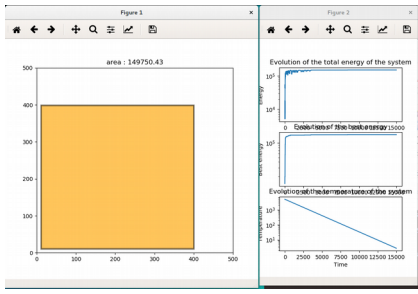
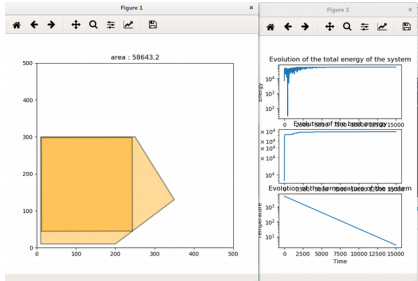
### 2. Tried algorithm : Simulated annealing and Particle swarm

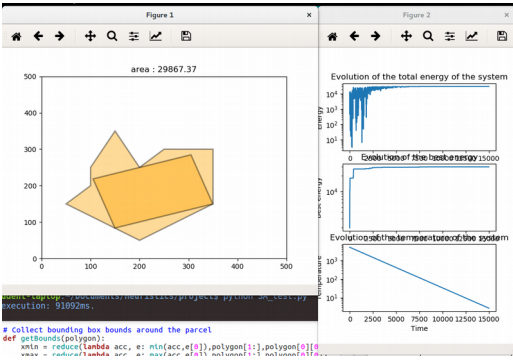
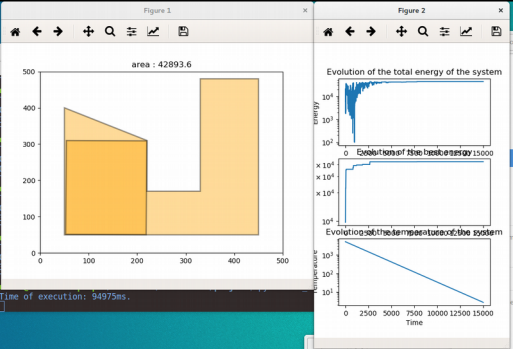
### 3. Compare results of the same problem set

\* Time complexity : Simulated Annealling :  $O(n^2)$  vs Particle swarm  $O(n^2)$

\* S-best: by average and standard deviation.

Because of TSP is a stochastic problem, I ll take the average of 3 S-best (best solution) provided by simulated annealling and Tabu search. Then compute the standard deviation for each number of cities.

Problem size	Simulated Anealling result T0 = 5000; IterMax = 150000	Tabu search result Nb_cycles = 500 Nb_particle = 20
1 <sup>st</sup> polygon	 <p>Mean: 149134 Std = 638.17</p>	<p>Psi = 0.3; Cmax = 1.3 Mean:92141 Std: 28249.85 results are very bad so we change other params</p>
2 <sup>nd</sup> polygon		<p>Psi = 0.7 ; Cmax = 1.7 Mean:58139 Std:561.3</p>

	<p>Mean: 58722.3 Std = 366.6</p>	
3 <sup>rd</sup> polygon	<div><p>Figure 1 shows a 3D plot of a yellow polyhedron with area 29867.37. Figure 2 shows three time-series plots: 'Evolution of the total energy of the system', 'Evolution of the heat energy', and 'Evolution of the temperature of the system'. The temperature plot shows a linear decrease from 10<sup>1</sup> to 10<sup>-1</sup> over 15000 time units.</p><pre># Collect bounding box bounds around the parcel def getBounds(polygon):     minx = reduce(lambda acc, e: min(acc, e[0]), polygon[1:], polygon[0][0])     maxx = reduce(lambda acc, e: max(acc, e[0]), polygon[1:], polygon[0][0])     miny = reduce(lambda acc, e: min(acc, e[1]), polygon[1:], polygon[0][1])     maxy = reduce(lambda acc, e: max(acc, e[1]), polygon[1:], polygon[0][1])     return (minx, maxx, miny, maxy)</pre><p>time of execution: 91892ms.</p></div> <p>Mean: 29729 Std: 125.677</p>	<p>Psi = 0.7 ; Cmax = 1.7 Mean: 29256.67 Std: 1047.69</p>
4 <sup>th</sup> polygon	<div><p>Figure 1 shows a 3D plot of a yellow L-shaped polyhedron with area 42893.6. Figure 2 shows three time-series plots: 'Evolution of the total energy of the system', 'Evolution of the heat energy', and 'Evolution of the temperature of the system'. The temperature plot shows a linear decrease from 10<sup>1</sup> to 10<sup>-1</sup> over 15000 time units.</p><pre>time of execution: 94975ms.</pre></div> <p>Mean : 43982.6 Std: 1512.122</p>	<p>Psi = 0.7 ; Cmax = 1.7 Psi = 0.9; Cmax = 1.9 → very bad Mean: 35504.3 Std: 4814.5</p>